Софийски университет “Св. Климент Охридски”
Факултет по математика и информатика

Дипломна работа

Складова и документираща система за нуждите на производство на електронни апарати

Бисер Емилов Грозданов
Магистърска програма: “Разпределени системи и мобилни технологии”
Катедра: “Информационни системи”
Фак.№: M21414
Научен ръководител: доц. Силвия Илиева

Съдържание

1. УВОД

Съвременният компютърен свят е неограничен. Непрекъснато се появяват различни платформи и нови разнообразни технологии. Все повече се увеличава и зависимостта на обществото от компютърната техника и високите технологии. Налаганите постоянно стандарти за качество изискват по-стриктно водене на документации и изготвяните преди, без връзки помежду си, бланки и документи, стават все по-трудни за справки и поддържане. Това подтиква разработката на всякакъв вид приложения, които да улесняват ежедневния живот на хората като ги превърнат в потребители на тези приложения.

1.1. Цели и обхват на разработката

Настоящата дипломна работа представлява система, разработена специално за нуждите на фирма "Боуви-България" ЕООД. Фирмата се занимава с производство на електронни апарати, главно за медицината. Дъщерна компания на водеща американска фирма в областта, известна с имената AARON MEDICAL и Bovie Medical Industries, "Боуви-България" ЕООД е създадена, за да се изнесе част от производството на гореспомената фирма извън граница. Дейността на фирмата се изразява във внасяне на компоненти от фирмата майка, по така наречения митнически термин - „за активно усъвършенстване" и производството на готово изделие от тях, което се изнася обратно към USA. При необходимост или икономическа изгода част от компонентите могат да се внесат по редовен внос или да се закупят в България. За да се спазват законовите изисквания за внесените за активно усъвършенстване елементи, е необходимо те да бъдат изнесени обратно от страната във вид на готово изделие, най-късно една година след като са внесени или да бъде представен протокол за брак. Това налага тези елементи да бъдат третирани с приоритет при износа, т.е. при наличие на елементи внесени за активно усъвършенстване и такива внесени с редовен внос или закупени в България, с приоритет се изнасят тези, които са по линия на активно усъвършенстване.

Фирмата е сертифицирана и по ISO9001, което налага воденето на много стриктна документация. Отделно изискванията на Bovie Medical Industries са да се работи с конкретни номенклатури и рецептурници, които трябва да бъдат въведени в системата.

Набора от документи, които се използват са:

- Митническа декларация за активно усъвършенстване;
- Митническа декларация за редовен внос;
- Митническа декларация за износ по активно усъвършенстване;
- Митническа декларация за редовен износ;
- Митническа декларация за унищожение;
- Проформа фактура;
- Поръчка за доставка;
- Складова разписка;
- Протокол за качество;
- Протокол за брак;
- Приемо-предавателен протокол;

- Американска Фактура;
- Българска Фактура.

Част от тези документи се изискват от Митническите власти на страната, друга част - от Bovie Medical Industries и от изискванията по ISO9001.

За да бъде създадена необходимата функционалност, обхващаща бизнес процесите на Боуви-България ЕООД, трябва да се изпълнят следните задачи:

- Да се изследват и опишат използваните документи, данни и бизнес процеси;
- Да се опише и създаде база от данни със съответната структура на таблиците, релации и ограничения;
- Да се създаде система, която реализира необходимите функционалности за осъществяване на поставените цели. Най-общо тя обхваща:
    1. Създаването на интерфейс, който реализира въвеждането на всички необходими основни данни;
    2. Реализирането на система за сигурност – вход в системата с потребителско име и парола, различни права за различните потребители и т.н.;
    3. Създаването на функционалност за въвеждането, поддържането и съхранението на съответните документи;
    4. Изготвянето на необходимия брой справки, които да придадат бизнес стойност на използването на системата.

1.2. Полза от реализацията

Сладовата и документираща система има за цел да обхване създаването на всички описани в горната подточка документи. Макар че се въвеждат цени в съответните валути и се следят наличностите като суми в склада и извън него, пишат се фактури и поръчки, системата няма за задача да обхване процеса счетоводно. Главните цели са да се подпомогне контролирането на производството и проследимостта на елементите, да се улеснят процесите по внос на първичните и износ на готовите компоненти, проследяването им, както и да се автоматизира планирането на снабдяването на склада с компоненти и материали.

Допълнително трябва да се защити достъпа до системата на Боуви-България ЕООД, с потребителско име и парола, както и да се осигури възможност за наличието на различни права за работа с нея. Например на някои от потребителите трябва да им бъде позволено единствено да могат да пускат справки без да имат права за промени или създаване на документи. От друга страна системата трябва да дава възможност за печат на документите и справките, както и да бъде гъвкава и сравнително лесно да се разширява и актуализира.

Разбира се без наличието на удобен и лесен за усвояване потребителски интерфейс, горните цели няма как да бъдат изпълнени.

1.3. Структура на дипломната работа

В първа глава са встъпителни думи. Описана е необходимостта от системата, нейния обхват, задачи и цели, както и практическата полза от нея.

Във втора глава е направен кратък преглед на SAP R/3: Какви са възможностите да бъде използван през WEB клиент. Как технически се прави това и какви са предимствата, и недостатъците. Направено е и кратко функционално сравнение между предлаганите

възможности на модула в SAP за материално стопанство и тези на складовата и документираща система.

В тази глава са описани и общи характеристики на избрания модел за разработка, разгледани са езика за създаване на WEB приложения – PHP, както и използваната база от данни – MySQL, какви са предимствата им и защо е избрана реализация с WEB сървър – Apache, PHP и MySQL.

В следваща подточка са разгледани моделите за разработване на софтуер – прототипен модел, модел с повторно използване на софтуер и модела за разработване с участието на потребителя. Разгледани са именно те, защото тяхното комбиниране е използвано при разработването на складовата и документираща система. Обяснен е подхода за създаването и, който е илюстриран и с блок диаграма.

В трета глава има 3 подточки. В първата с подробност са разгледани функционалните изисквания към системата. В следващата подточка е показан самия функционален модел под формата на диаграма на потребителските случаи (Use Case Diagram), разработена на Rational Rose, а в последната - модела на базата данни с връзките между таблиците, разработен с помощта на Clay добавката за Eclipse.

В четвърта глава е самата реализация на системата. Тя се състои от две подточки. В първата са дефинирани всички таблици в системата. Те са представени в табличен вид, като за всяко поле име описание. Отделно за всяка таблица са засегнати най-важните моменти с кратък текст.

Втората подточка е самата PHP реализация. Тя формално е разделена на пет подточки – елементи, документи, справки, потребители. В нея описателно е обяснен алгоритъма на всички разработени функционалности и е засегнато е използването на по-важните функции в тях, като самия код на тези функции е даден в приложенията (глава 9). Отделно, за всяка функционалност са изредени референтните файлове, които я реализират.

В главата тестване и внедряване накратко е обяснено какво представлява качествен софтуер, разгледани са различни видове тестове и това как те са използвани в реализираната система. Има три подточки. Първата е за това как е осигурена коректността на реализацията. В нея е дадена и блок схема на подхода за гарантиране на тази коректност. Втората подточка обхваща производителността на системата и как е тествана тя, а третата обяснява стъпките по вкарване в експлоатация и е описано как е реализирано първоначалното зареждане на данни.

Следващата глава е ръководство за потребителя. То няма за цел да обхване всички екрани и функционалности с обяснение на всяко поле за попълване и т.н., а да покаже различните възможности на интерфейса и как да се използват те.

В седма глава е направено кратко обобщение на разгледаните глави, засегнати са и възможностите за бъдещо развитие и разпространение на системата.

В осма глава са изредени цитираните в текста източници, както и използваните такива за придобиване на необходимите знания за реализирането на складовата и документираща система, а в девета глава са дадени програмните кодове на по-важните функции, използвани при реализирането на системата.

2. Използвани модели, спецификации и технологии

В главата ще бъдат разгледани, аспекти на модулите Материално стопанство и Логистика на системата SAP R/3, предлагащи подобна функционалност на разработваната. Ще бъде разгледано и начина на реализиране и използване на SAP системата през WEB интерфейс.

Ще се обърне внимание и на различните технологии и средства, с които е реализирана системата. Какви алтернативи и варианти съществуват.

2.1. ERP системи - предимства и проблеми

В тази част ще бъде разгледана общо системата SAP R/3 както и по специално използването на модула Материално стопанство. Ще бъдат посочени предимствата и недостатъците от използването й. Избрана за разглеждане е SAP система, защото според различни публикации [1], в момента SAP обслужва повече от 32000 организации в повече от 120 страни в света, като клиентите и непрекъснато растат.

Пазарния дял на SAP представлява 30% от софтуера за финансово управление, 24% от този за човешките ресурси и 34% в областта на производството. Това го прави безспорен лидер в областта на ERP системите.

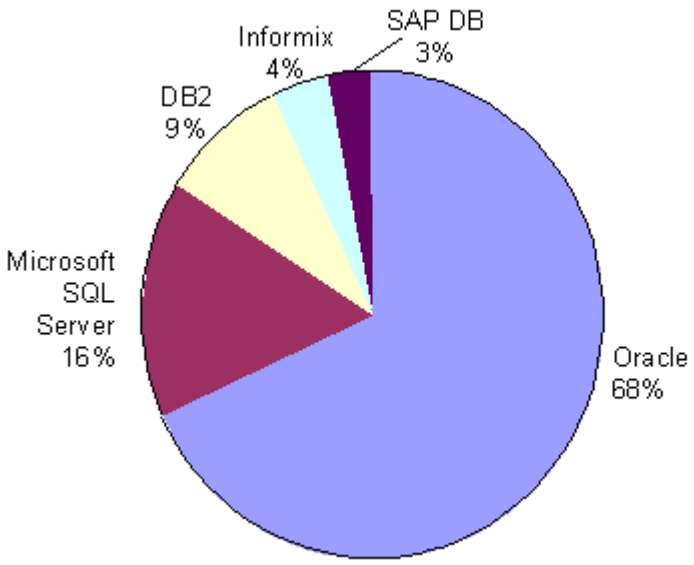
Бъдещите планове на компанията са да се обхванат не само големите компании, а системите им да стават все по достъпни и за малки и средни предприятия, но за сега този момент още не е дошъл.

Разработени са много специални модификации на SAP системата за различни видове индустрии (например за рафинерии и нефтени компании), както и за публичния сектор. Голямата гъвкавост, която SAP предлага, има своя цена. При закупуване на система трябва да се вземат под внимание не само цените на лицензите и тези за необходимия хардуер, но и не по-малко средства от споменатите - за настройване и интегриране на системата с другите налични приложения в компаниите. Сложността и новите принципи заложени в него затрудняват средно статистичните потребители и те имат нужда от много обучения и време за адаптация.

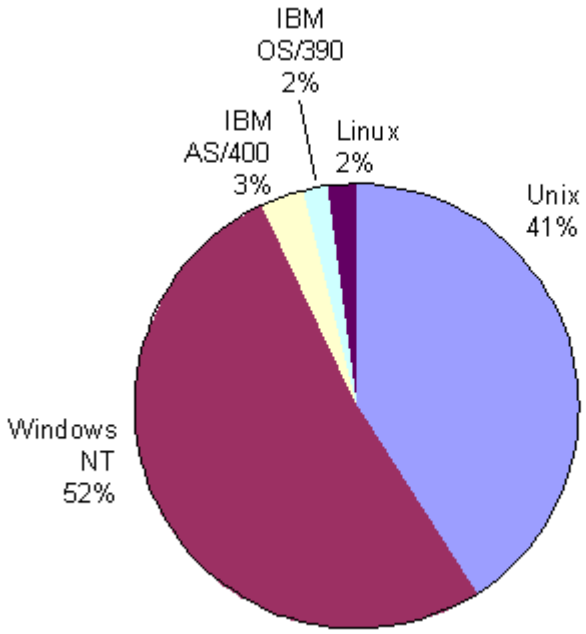
Ако има една дума, която обединява R/3 потребителите, тя е “заети”. Увеличаването на изискванията и подобренията държат всички потребители заети и непрекъснато ангажирани в процеса. В средите на клиентите на системите, предлагани от SAP, е разпространена фразата : „Хванал ли си се със SAP, това е до живот.”

За разпространението на SAP в България водеща роля играе това, че големи наши компании са вече собственост на чужди фирми, които използват централизирано SAP и налагат използването й и в тукашните си филиали. Такива примери са „Златна Панега”, „Соди Девня”, „Нефтохим” и много други. Обикновено сървърите и съответно администрирането им се намират в страната, от която е компанията. С добрите комуникации това вече не е проблем, а способства за лесно централизиране на данните и управление на огромните дружества. Това улеснява и процеса на администриране и поддръжка и е един от големите плюсове за внедряване на една и съща система в цялата компания.

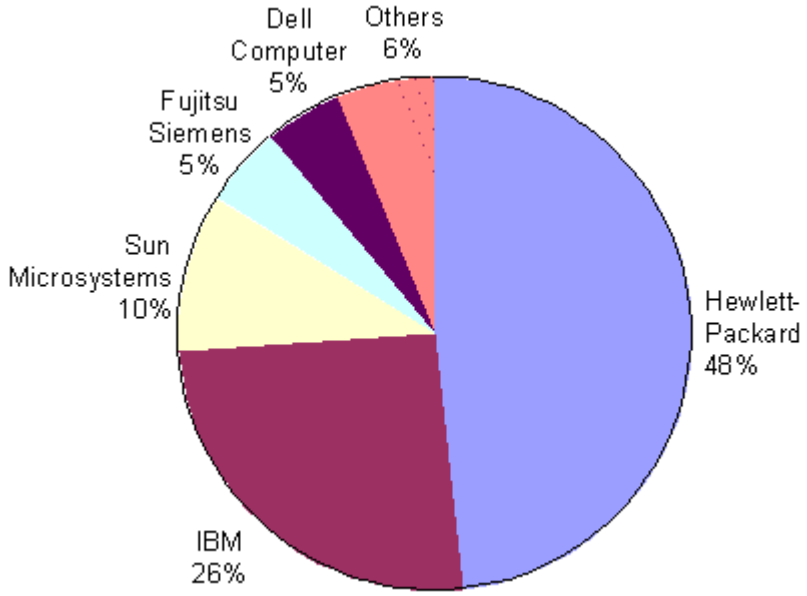
При започването на проект на SAP базирана информационна система от съществена роля е избора на операционна система и база от данни за нея. Изборът обикновено е ограничен до списък, предоставян от SAP - какви платформи (комбинации от операционна система и база от данни) се поддържат за съответната версия на инсталирания продукт. Този списък е много широк, но в него обикновено не влиза свободен или софтуер с отворен код, което още веднъж оскъпява решението и свива възможния избор на платформи. Разбира се, при достатъчно опитни хора, занимаващи се с поддръжката и инсталацията, е възможно да бъде инсталирана и на дистрибуции, близки до поддържаните, но тогава автоматично отпада поддръжката от страна на SAP, което за системи от подобен ранг е изключително опасно и необосновано, и със сигурност ще излезе финално по-скъпо отколкото да се плати за лицензи за поддържана платформа. Следващите графики показват процентно използването на различни операционни системи, бази от данни и хардуер със SAP системи – фиг. 2.1.1, фиг. 2.1.2 и фиг. 2.1.3.



Фиг. 2.1.1. Разпределение на SAP пазара за бази данни

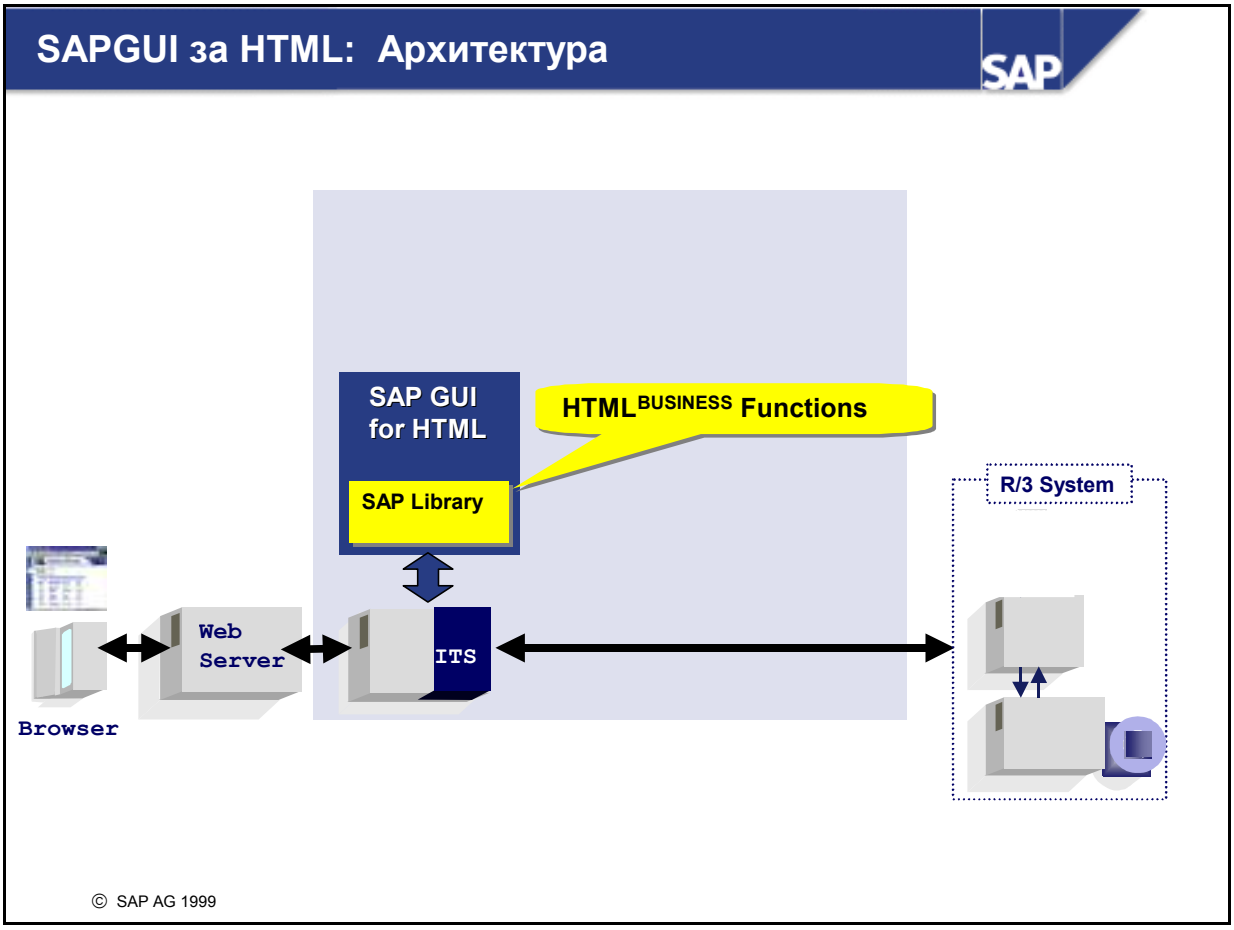


Фиг. 2.1.2. Разпределение на SAP пазара за операционни системи:



Фиг. 2.1.3. Разпределение на SAP пазара за хардуер:

Стандартно в SAP системите начина на комуникация между работната станция и сървъра става посредством SAP GUI (Graphic User Interface), на базата на близък до TCP/IP протокол. Това осигурява достатъчно добър и бърз интерфейс за работа със системата като необходимите мрежови ресурси (около 5 Kb за сесия) са много малки в сравнение с възможностите, които дават сега локалните мрежи и Интернет. От гледна точка на разработваната WEB базирана система трябва да кажем , че и SAP разполага с възможност за достъп до Приложния сървър посредством WEB клиент. Това става по схемата, показана на фиг. 2.1.4:



Фиг. 2.1.4. Архитектура на SAPGUI за HTML

По тази схема SAP транзакция може да се стартира, използвайки всеки WEB клиент и не е необходимо потребителите да инсталират GUI. Вместо това SAPGUI за HTML е инсталирано на ITS. Тази инсталация включва библиотеката на HTMLBUSINESS functions, която е необходима за динамичното генериране на HTML страниците. След което тази генерирана HTML страница се изпраща към WEB клиента. Цикълът на заявка-отговор между SAP Системата и WEB клиента е структуриран както следва:

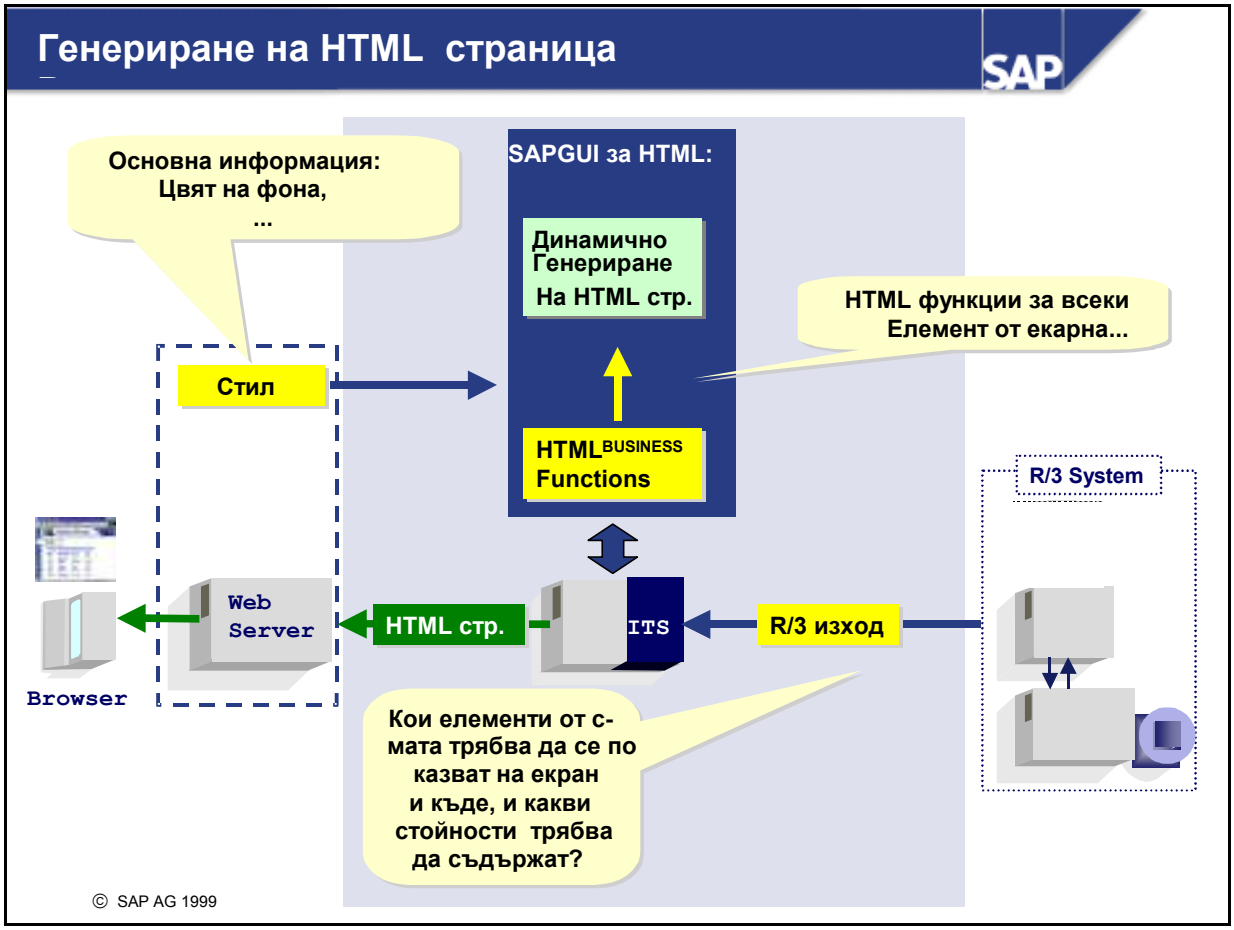
1. Заявката идва от WEB клиента към Web сървъра, който изпраща заявките към ITS.
2. ITS се уверява, че потребителят е влязъл в SAP системата.

3. Данните, които нормално се изпращат към презентационния сървър, когато е зададен екрана, се изпращат към ITS.
4. ***SAPGUI за HTML*** генерира HTML страница и я изпраща към Web сървъра.
5. Web сървърът изпраща HTML страницата към WEB клиента.

За да генерира HTML страница, системата се нуждае от следната информация:

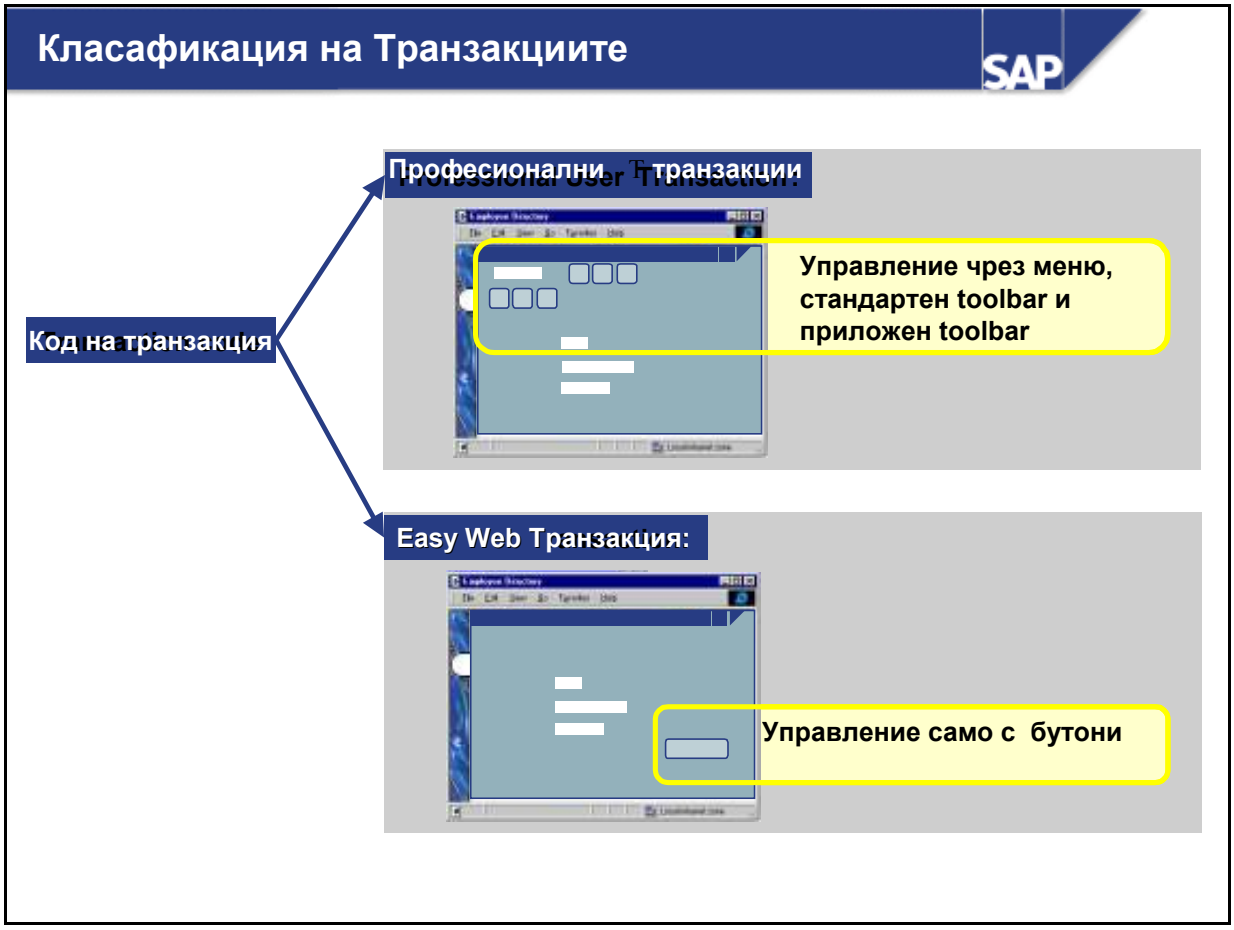
1. **R/3 изход:** Информацията, която нормално би била изпратена към презентационния сървър, се изпраща към ITS.
2. **Основна информация:** Съхранявана на Web сървър като *стилове*.
3. **SAP Библиотека:** HTML^{BUSINESS} functions, необходими за функциите на елементите от различните екрани, се съхраняват в библиотека на ITS.

Процесът е илюстриран на фиг. 2.1.5.



Фиг. 2.1.5. Генериране на HTML страница

В SAP е предвидено да съществуват два вида WEB транзакции. Едните са така наречените „Професионални транзакции", а другите - „Easy WEB" транзакции. Виж фиг. 2.1.6.

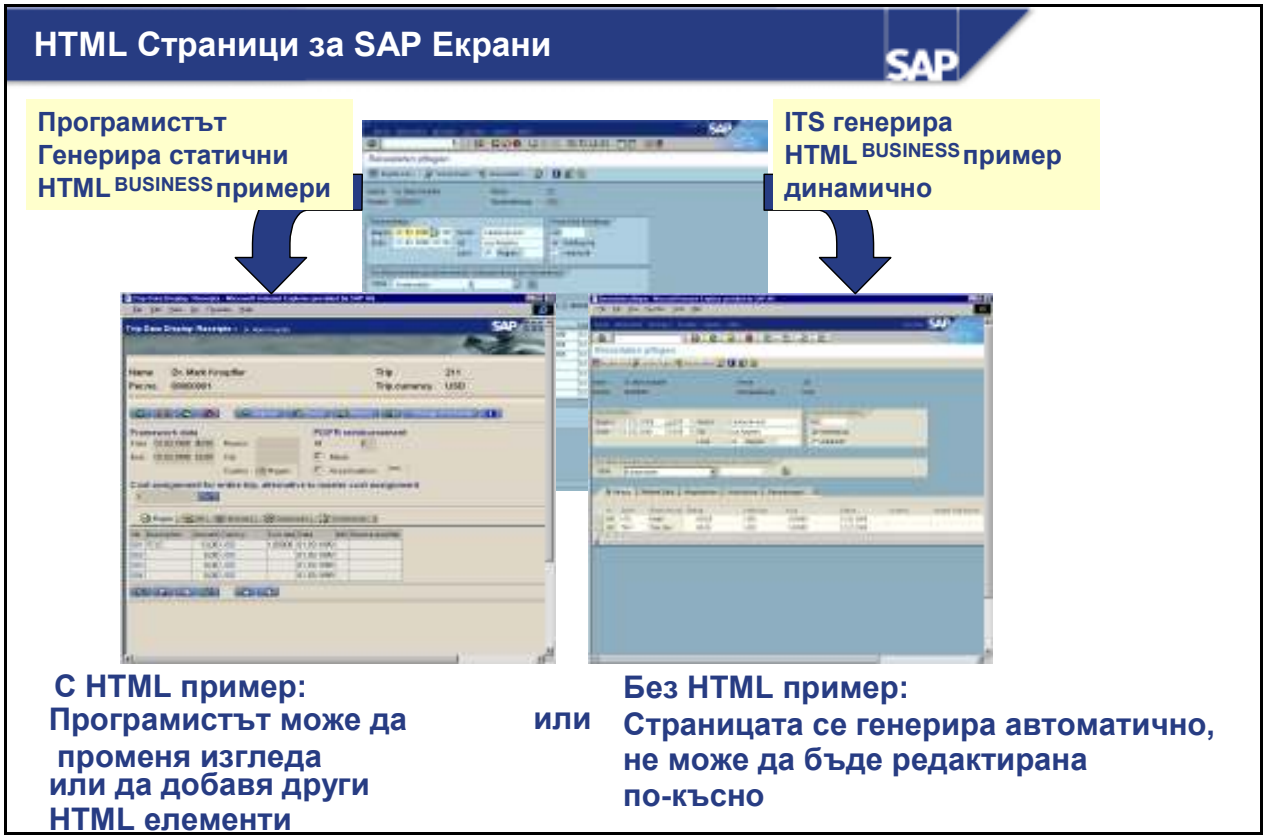


Фиг. 2.1.6. Видове WEB транзакции в SAP

Още при създаването на транзакцията трябва да се специфицира дали тя е „професионална" или „Easy Web". По подразбиране е Professional User Transaction. Тя трябва да остане такава ако потребителите желаят комплексни навигационни функции.

Easy Web Транзакциите удовлетворяват изискванията на „случайните" потребители или неопитните такива, извикващи транзакции през Web. Управлението е възможно най-просто. Потребителят трябва да може да изпълнява всяка функция без използването на меню, стандартен ред с бутони или приложен (специфичен) ред с бутони.

От гледна точка на програмистите съществуват два подхода при създаването на HTML страниците. Единият е да се генерират автоматично от ITS, а другия да се генерират от програмиста чрез използването на HTML шаблони. Процесът е илюстриран на фиг. 2.1.7.



Фиг. 2.1.7. Начини за създаване на HTML страници.

1. ***SAP GUI за HTML* генерира HTML динамично** от съответните SAP екрани:

Може да се избират графични атрибути, използвайки стиловете (като шрифт, размер, цвят). Иначе елементите от SAP екрана съответстват 1:1 към полетата на генерираната HTML страница. Никакви други HTML елементи не са валидни и никаква друга допълнителна разработка не е необходима.

2. **Изгледът на Web приложението може да бъде адаптиран,** използвайки **статични HTML BUSINESS примери**

Използването на статични HTML примери и MIME обекти позволява да се разшири изгледа според нуждите. С тези техники могат да се добавят повече бутони и картинки, дори и ако няма място за тях в примера. Полетата, които съдържат данни по подразбиране, могат да бъдат скрити. Също така могат да бъдат добавяни и функции. За сметка на това, създаването им е по-сложно, защото трябва да се осигури допълнителна по-нататъшна поддръжка.

От посочения начин и организация на използване на SAP система през WEB интерфейс става ясно, че има много особености и не е лесна задача нито за програмистите, нито за администраторите да го осъществят. Освен това при положение, че стандартна инсталация на последните версии на SAP GUI не изисква повече от 80 MB дисково пространство и за да върви то достатъчно добре, не е необходимо повече от 128 MB RAM, начина на използване през WEB интерфейс все още не е много популярен.

Модули MM и Логистика в SAP

В тази част накратко ще бъде разгледана функционалността, която дава системата SAP R/3 в частта и за материално стопанство (тази част, която засяга близка функционалност до тази, предлагана от изградената складова и документираща система - тема на тази дипломна работа). Също така ще бъде направен опит за сравнение между системите като предлагана функционалност, цена и време за реализиране на решението. Ще бъде направено и кратко обяснение на използваните термини, които са характерни за SAP.

Основни термини

В системата SAP R/3 заводът (plant) е централна организационна единица в логистичните модули (т.е. и в модул Материално стопанство), служеща за подразделянето на една организация (ведомство, предприятие) съобразно нуждите на производството, снабдяването, планирането на материалните потребности, поддръжката и ремонтите. Той може да бъде мястото, където протичат стопанските процеси, респективно мястото, за което се доставят/закупуват материали.

В изгражданата система за Боуви-България не е предвидена възможност за работа на повече от един „завод", защото това не е задача по заданието, но само по себе си този факт я прави по-малко гъвкава и разширяема. Например, ако в бъдеще възникне необходимост от разделяне на документооборота, складовете и отговорностите по производството и снабдяването, тогава ще трябва да се правят допълнително промени по програмния код и базата данни.

В SAP R/3 складът (storage location) е организационна единица, позволяваща разграничаването (групирането) на количествата от материални запаси в рамките на даден завод.

В складовата и документираща система не е предвидена възможност за създаване на складове. По задание склада е един. Всички други местоположения са организирани като външни за системата, но в справките те могат да се обединяват и по този начин да се обобщават материалите от тях. По надолу във функционалното описание е обяснено как е организирано и изпълнено това.

Основните записи на материалите в SAP R/3 се създават като се избират промишлен отрасъл и вид на материала. След това се генерира уникален номер за системата.

В складовата и документираща система е изградена подобна архитектура, но заради спецификата на производството, за което е предназначена, разделението е по група и подгрупа на компонентите. Например, за компонент с номер 00-000-000-C, първите две цифри са за групата му (например кондензатори или пластмасов детайл и т.н.), следващите три са за подгрупата на компонента (например електролитен), а последните три цифри са идентификация на конкретния компонент (например колко фарада, ома и т.н.). Буквата определя ревизията на компонента.

Основни функционалностти

В системата SAP R/3 функционалността по прогнозиране на материалите (Material Forecast) използва различни математически модели, както и данни от минали периоди за предвиждане (прогнозиране) на бъдещи стойности, като основните прогнозни модели са константен модел, тренд-модел, сезонен модел и сезонен тренд-модел. Този подход е приложим за канцеларски материали или за производство, което е със сравнително

постоянна гама продукти. Не такъв е случаят обаче с "Боуви-България" ЕООД. Тук производството е силно зависимо от конкретните поръчки и при липса на такива може да не се произвежда нищо. Цялата продукция се изнася и предава на Bovie Medical Industries – USA. Това налага различен подход при изграждането на функционалността за планирането. Тя се базира на поръчките на възложителя (какво да бъде произведено), на поръчките за доставка на материали, които вече са били направени и на текущото състояние на склада за компоненти и материали. По-надолу в изложението по-подробно ще бъде обяснен процеса на планиране.

Функционалността по оценка на доставчици (Vendor Evaluation) е част от подкомпонента Снабдяване в SAP R/3. Оценката на доставчици подпомага снабдителите като оптимизира процеса на доставки. Оценката на доставчици опростява процеса по избор на източник на доставка и прави възможен постоянния контрол и преразглеждане на взаимоотношенията с доставчиците. Тя се извършва на базата на унифицирани критерии. Организационното ниво, на което се извършва оценката на доставчиците, е снабдителската организация.

При оценката на доставчици потребителите използват точкова система (от 1 до 100 точки) като дейността им се преценява според четири главни критерия.

Крайният резултат дава на служителите, ангажирани в снабдителския процес, обща представа относно доставчиците и позволява извършването на сравнителни анализи за тях.

Главните критерии за оценка на доставчици в стандартната система R/3 са:

- Цена;
- Качество;
- Срокове на доставка;
- Качество на услугата.

При необходимост могат да бъдат дефинирани до 99 главни критерия. Потребителят може да променя влиянието на индивидуалните критерии спрямо общия резултат. За всеки основен критерий може да бъде зададено тегло (тегловен коефициент). Общата оценка за доставчика се изчислява като резултат от претеглените точки за основните критерии.

Всеки главен критерий може да бъде разделен на няколко подкритерия, което позволява по-диференцирана и детайлна оценка. За подкритериите също е възможно да се дефинират тегловни коефициенти.

Системата SAP R/3 предоставя пет подкритерия, които по правило са достатъчни за целите на оценката на доставчици. При необходимост потребителят може да дефинира до двадесет собствени подкритерия.

Резултатите по отношение на подкритериите могат да се изчисляват по няколко начина:

- автоматично;
- полуавтоматично;
- ръчно.

При автоматичното изчисление резултатите се изчисляват на базата на данни, които вече съществуват в системата.

При полуавтоматичното изчисление потребителите въвеждат индивидуални резултати за някои по-важни материали, а впоследствие системата изчислява резултата на по-високото ниво на обобщение.

Вижда се, че SAP R/3 не може да бъде достигнат лесно в това отношение от индивидуална система, като разработваната например, чиято главна цел не е този аспект на логистиката, защото се работи с краен брой фирми доставчици, които са силно специфицирани и предлагат почти уникални услуги на пазара. Поради тези причини това не е поставено като задача при проектирането и реализирането на складовата и документираща системата на Боуви-България ЕООД.

В системата SAP R/3 възникналите потребности от материали се отразяват с вътрешни за организацията (ведомството, предприятието) документи, наречени заявки (искания) за доставка. Те са основата за всички последващи дейности в процеса на снабдяване.

Поръчката за доставка (purchase order) е формалната заявка или инструкция от страна на снабдителската организация към даден доставчик, за доставка на желано количество материали в определен период от време.

Другите най-важни документи, в частта на логистиката и материалното стопанство, са приемо-предавателния протокол и складовата разписка, които служат съответно за предаване на материали към доставчик или МОЛ и заприхождаване на изделия след поръчка за доставка.

В разработваната система принципа е същия като не е предвидено наличието на искания за доставка поради централизираното определяне на необходимостите от материали в "Боуви-България" ЕООД, което се обуславя и от големината на производството.

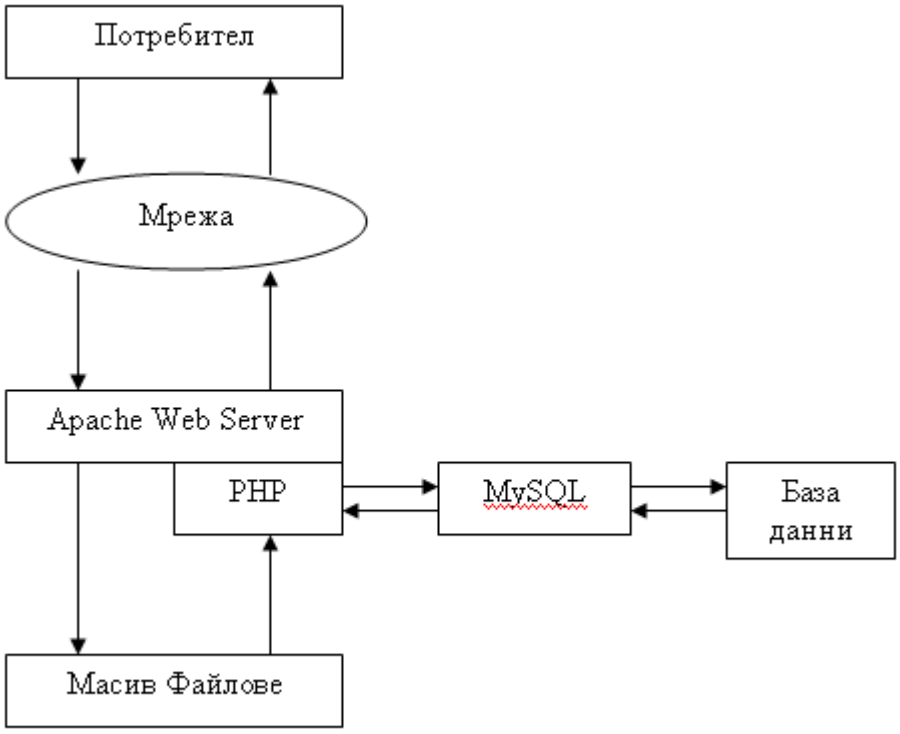
Изводи:

От направените сравнения на складовата и документираща система с модула на SAP R/3 за материално стопанство и логистика, става ясно, че системата не е конкурент на R/3 по отношение на предлагана функционалност, възможности за настройка и адаптация. В нея липсват функционалности като например тази за задаване на оценки за доставчик, но за сметка на това е много по-лека и евтина за поддръжка. Разработена специално за целите на "Боуви-България" ЕООД, тя не предлага гъвкави механизми за настройка и връзки с други модули, но обхваща цялата необходима функционалност за фирмата и подлежи на лесно разширяване при необходимост. Липсата на големи изисквания за хардуера също спестяват десетки хиляди от закупуването на мощни сървъри.

2.2. Избор на програмен език, среда за разработка и база от данни. Помощни средства.

Складовата и документираща система е реализирана като WEB клиент-сървър базирана система. При тези системи приложението работи като съвкупност от WEB форми, които се извикват в Интернет клиента на потребителя. Голямо удобство на този вид системи е, че не се изискват инсталиране на специална програма "клиент" при всеки един потребител.

Механизмът, по който работи WEB-приложението е показан на фиг. 2.2.1.



Фиг. 2.2.1 Механизъм на работа на системата

За среда на разработката е избрана MySQL Database Version 5.0 - база от данни и основен скриптов език за разработка PHP 4.4 (hypertext preprocessor). Както във всяка малко по-сложна WEB страница, за контрол на поведението на WEB клиента се използва JavaScript.

За създаването на менютата, бутоните и по-голяма част от визуалните елементи е използвана XARA WEBSTYLE 4. Това е инструмент за използване на шаблони за създаване на менюта, бутони, картинки и т.н.

Основните предимства на модела, показан по-горе, по който е реализирана системата е, че всички елементи от нея се разпространяват безплатно. В частност на сървъра е инсталирана операционна система Debian GNU Linux, която също е безплатна. Със същата лекота може да се инсталира каквато и да е операционна система.

Проектирането на базата данни е направено с помощта на добавка (plugin), за Eclipse SDK, наречена clay, а създаването и с помощта на phpmyadmin, и посредством генерирания SQL код от clay модела. Тези инструменти също се разпространяват безплатно.

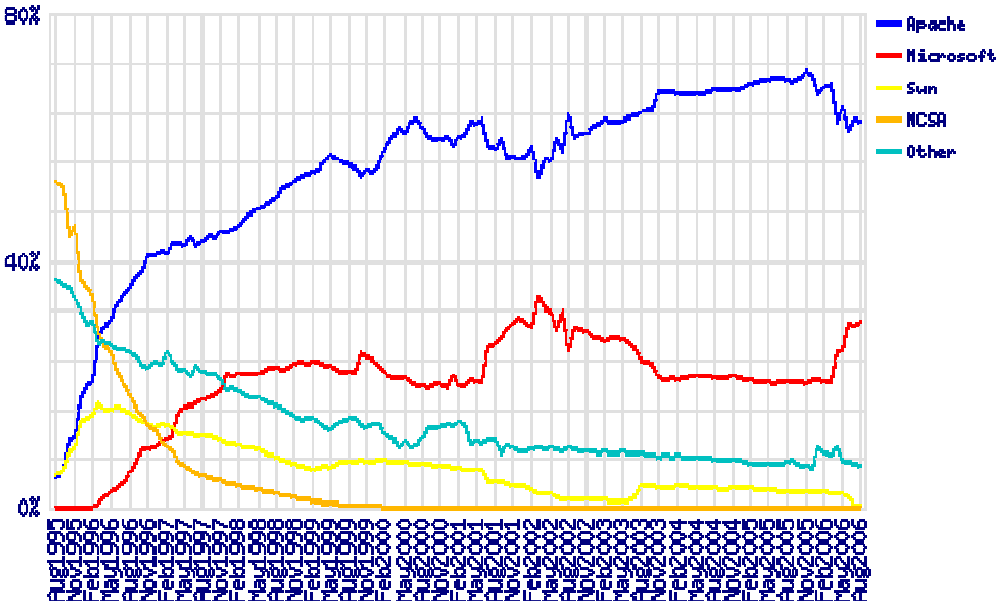
2.2.1. Език за реализация - PHP

Както подсказва името му това е език за динамично генериране на WEB страници (HTML - код). Начинът на обработка на PHP страници не се отличава съществено от обработката на HTML страници. Най-общо казано, добавена е една стъпка. Вместо да се изпраща статична HTML страница към потребителя, сървъра извършва определени

действия в зависимост от PHP кода: PHP ще вземе някои решения и ще създаде страница, която е подходяща за конкретната ситуация. Така при използване на PHP, действията са следните:

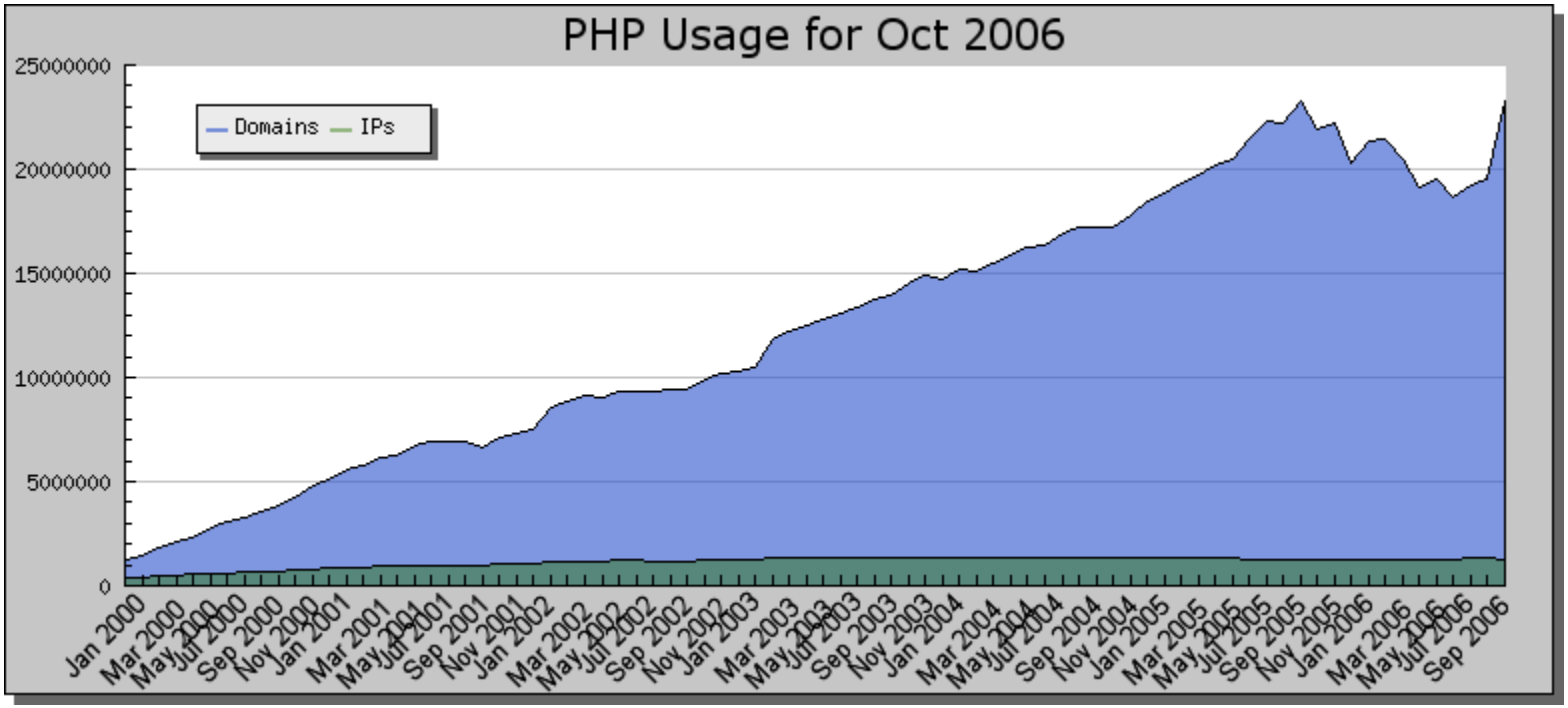
- Прочита се заявката на WEB клиента (browser).
- Намира се исканата страница на сървъра.
- Изпълняват се инструкциите, заложени в PHP, за да се модифицира страницата.
- Изпраща се чрез Интернет страницата обратно към WEB клиента.

PHP е избран за използване, защото е най-популярното средство за разработване на WEB приложения в последните години. Той предоставя много добри възможности за работа с база данни и масиви, което е от съществена важност за реализацията на системата. На следващите две фигури- фиг.2.2.1.1 и фиг. 2.2.1.2, може да се наблюдава използването на Apache и PHP през последните години.



Фиг. 2.2.1.1 Разпределението на пазара на WEB сървъри според Netcraft за периода Август 1995г. - Август 2006г.

PHP: 19,562,759 Domains, 1,305,799 IP Addresses
Source: Netcraft



Фиг. 2.2.1.2 Разпространение на PHP

Вижда се, че избрания подход е много популярен и следователно сравнително лесно може да се намерят специалисти за поддръжка на системата или за разширяването и.

Избирането на Apache като WEB сървър и самите възможности на PHP, предупределят избора на езика за разработка.

2.2.2. База от данни - MySQL

MySQL е започнала своето развитие по начин сходен на Linux, но в момента достига нивото на продукт, който е подходящ и за сериозни бизнес цели. Пример за това е факта, че най-популярната търсачка в Интернер – Google, използва MySQL технология.

MySQL е най-често използваната, в комбинация с PHP, база от данни. PHP има вградени функции, позволяващи извършването на различни операции, което е и една от причините тази база да е толкова популярна.

MySQL поддържа повечето функции, които се очаква от комерсиална база от данни, да поддържа. Транзакциите са съобразени с ACID модела, позволява създаването на индекси, поддържа стандартни типове данни, позволява репликиране на базите данни и др. Основната област, в която бяха слабостите на MySQL до скоро, е липсата на поддръжка на записани процедури (stored procedures) и тригери. Тези слабости са отстранени във версия 5.0 и сега MySQL е с пълен набор способности спрямо комерсиалните СУБД. Записаните процедури основно позволяват записването на SQL код на сървъра на базата данни и позволяват на потребителите и приложенията да извикват тези процедури със съответните им аргументи без познаване на самата SQL заявка. В допълнение на плюсовете като опростяване, благодарение на тази абстракция, системните администратори могат също да използват записани процедури с оглед подобряване на сигурността, тъй като е възможно да се дава достъп на конкретен потребител до набор от записани процедури, без да се дават права на потребителя над самата база данни. Друг голям плюс е модулът за запис на данни InnoDB, който поддържа компактен запис, използвайки 20% по-малко място. В допълнение дава възможност и за използване на транзакции, както и на свързани таблици посредством ключове и добавянето на ограничителни условия на тези връзки. Тези му възможности определят и избора му като тип на таблиците в складовата и документираща система.

За популярността на MySQL допринася и това, че за администрирането и има създаден много удобен и широко използван инструмент – phpmyadmin.

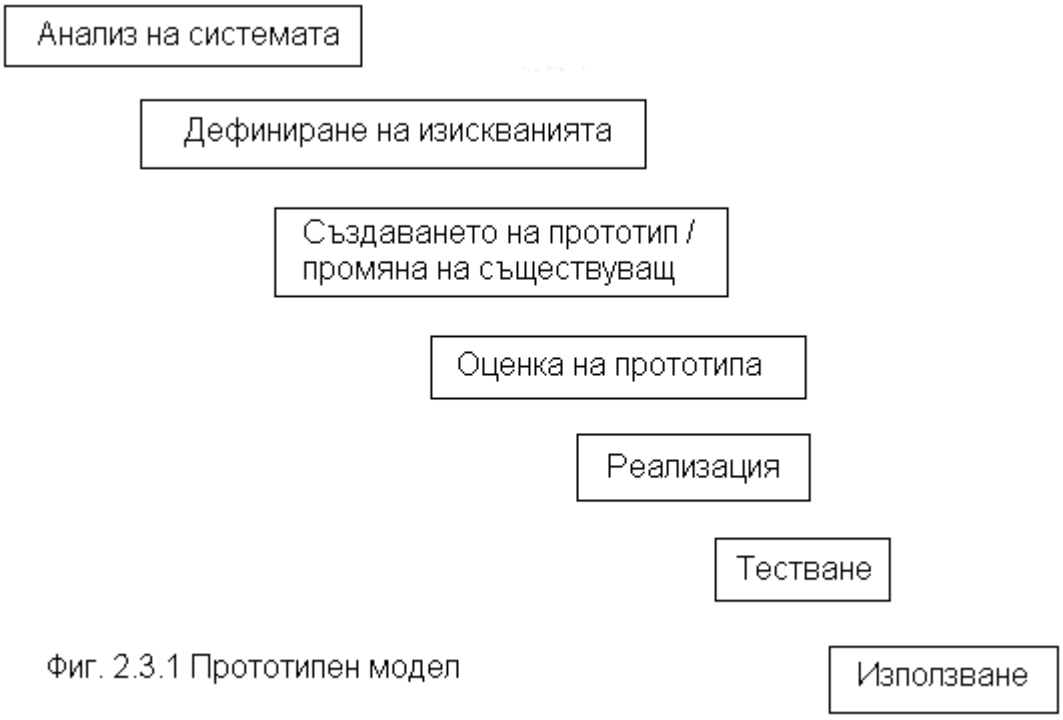
2.3. Модели за разработка на приложението

Използваният метод за разработка на приложението е комбинация от модела за прототипно изграждане на софтуер, този за повторно използване на софтуер (RE-USE) и разработване на софтуер с участието на потребителя. Какво представляват тези методи и как са използвани в изграждането на системата, ще бъде обяснено в тази точка.

Според [2] определенията за тези методи са:

Прототипен модел - модифициран хронологичен модел, който се използва интензивно в практиката. Същността му се състои в това, че се реализира умалена версия (прототип) на крайния продукт, върху която се правят експерименти за установяване на съответствие с изискванията на потребителя.

В зависимост от получените резултати се правят корекции с различна степен на връщане към предходните стъпки. Естествено, колкото по-близко е връщането, толкова по-бързо и евтино става окончателното завършване на прототипа. След окончателното доказване на правилността на функциониране на прототипа и съответствието му с изискванията за разработката се пристъпва към реализацията на крайния продукт. На фиг. 2.3.1 е илюстриран прототипният модел.



Фиг. 2.3.1 Прототипен модел

Както се вижда, след преминаването за първи път през първите 3 фази се достига до оценка на прототипа. Обикновено се установяват несъответствия с изискванията. Тогава разработчикът се връща на предходната фаза и променя прототипа (отначало чисто програмно, после на ниво проект, ако трябва), след което отново прави оценка. След известно повторение на този цикъл може да се окаже, че само промяна в прототипа не води до желаните резултати и тогава връщането трябва да стане една фаза по-назад. Следователно се налага промяна в определянето на някои изисквания. Това само по себе си е неприятно, защото всъщност изисква съгласието на потребителя (възложителя). Може обаче да се окаже, че дори и такива реални промени не водят до установяване на съответствие прототип-изисквания. Тогава следва връщане в начално положение, т. е. налага се да се направи наново изследване на проблема и възможностите за реализацията му, вероятно да се достигне до установяване на пропуснати особености и отново да се повторят всички следващи стъпки до оценката на новия прототип.

В даден момент прототипът отговаря на всички изисквания. Тогава се пристъпва към реализация на крайния продукт. Тестването, естествено, остава абсолютно необходима процедура и едва след неговото успешно приключване може да се пристъпи към използването на така създадения продукт.

Повторно използване (RE-USE)

Подходът за повторно използване (ПИ) е съвкупност от планирани и систематични дейности, насочени към максимално използване на съществуващи софтуерни елементи в процеса на създаване на нов софтуер.

В зависимост от формата и докъде се простира повторното използване, то може да бъде вертикално или хоризонтално.

При вертикално повторно използване в избрана приложна област се създават основни модели, които се прилагат във всички разработвани за тази област софтуерни системи.

При хоризонталното повторно използване се създават универсални компоненти, които могат да се вграждат в програмни системи за различни приложни области – например средства за управление на бази от данни, за разпределени среди, за изграждане на графичен потребителски интерфейс и др.

Разработване на софтуер с участието на потребителя

В зависимост от инициализацията си софтуерните проекти могат да се разделят на две групи.

Автономни са проектите без конкретен възложител. Обикновено до идеята за създаването им се достига след маркетингово проучване на състоянието и тенденциите на софтуерния пазар. Тези проекти се самофинансират от фирмите разработчици, като функционалните им възможности и изискванията, които трябва да удовлетворяват, се определят чрез изследване на съществуващи аналози (действащи или описани в литературата) и обобщаване мнението на потенциалните потребители.

Поръчани (възложени) са проектите с конкретен възложител, който финансира разработката и след завършването ѝ става неин собственик. Класическите подходи на разработване в този случай предвиждат участие на възложителя в началните фази (изследване, анализ на осъществяване) до дефиниране на изискванията и отново чак след окончателното му завършване – за оценка и приемане на готовия програмен продукт.

Основна идея на подхода на разработване с участието на потребителя е да се повиши активността и съответно отговорността на потребителя за качеството на програмния продукт, като той се включи активно и в някои междинни фази.

Може да бъде дадена следната дефиниция:

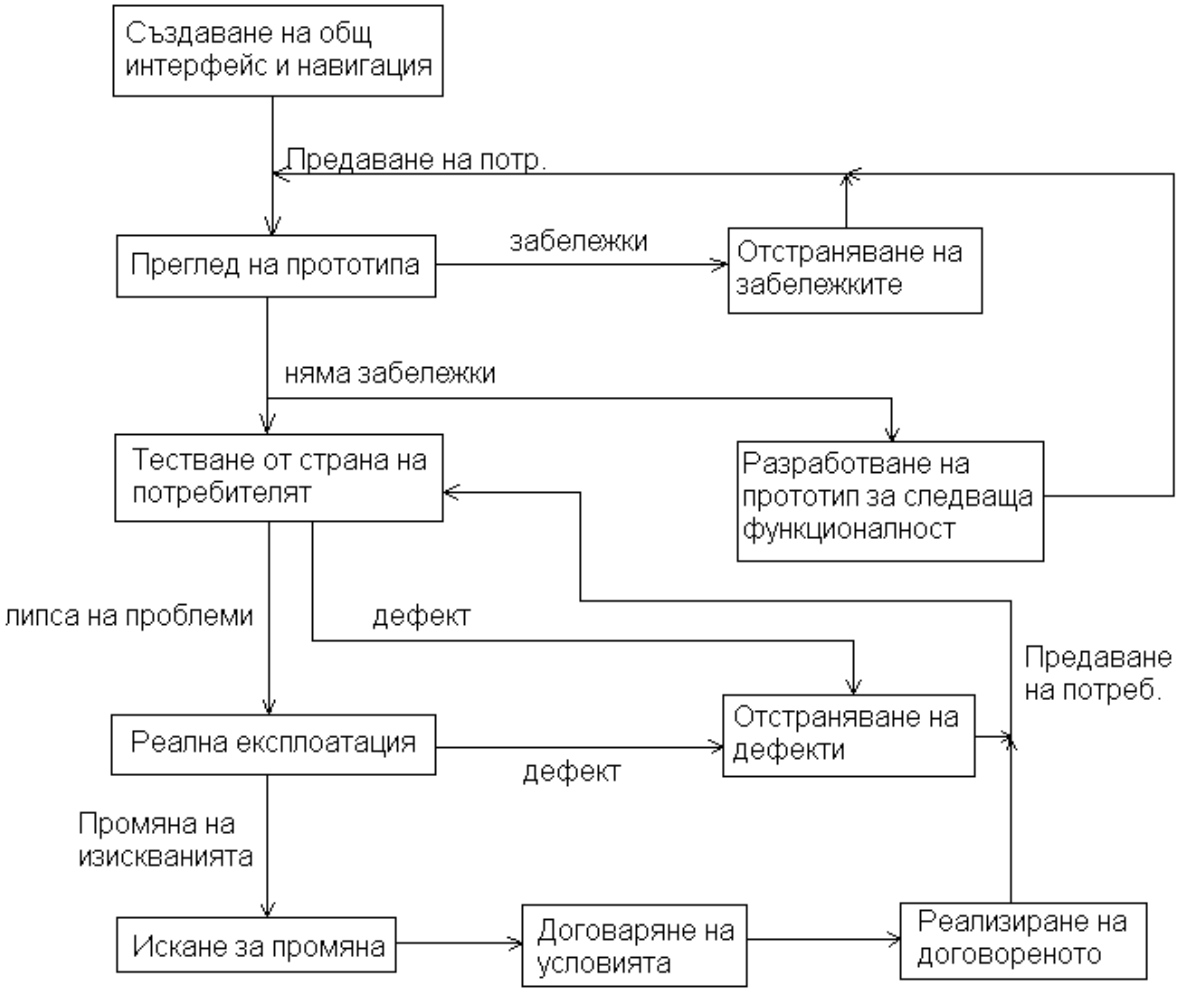
Разработване с участието на потребителя е съвкупност от методи, техники и целенасочени действия, осигуряващи активното участие на потребителя в проектирането и създаването на софтуерната система, която той ще използва.

Подходът се избира в началото на възложен проект и се следва систематично и планирано. Предполага се активно участие на потребителя. Осъществява се не само регистриране на реакциите му при работа с междинни версии на програмния продукт, а включването му в процеса на вземане на решения в критични за проекта ситуации.

Реализацията на подхода изисква специални техники на разработване. Една възможна и проверена в практиката техника е еволюционното прототипиране, при което разработването на ПП се разглежда като създаване на последователност от междинни ПП, всеки от които е разширение на предишния. След завършване на един прототип се извършва едновременно тестване на готовия прототип в реални условия (от потребителя) и създаване на следващия прототип (от разработчика). Освен ранно откриване на грешки

използването на прототип в реална потребителска среда позволява формулирането на допълнителни изисквания.

В складовата и документираща система подхода е следният. Чрез помощни средства за повторно използване на софтуер, са създадени общия интерфейс и навигация (хоризонтално и вертикално меню). След одобрението им от страна на потребителя, се създават първи прототипи на важните функционалности, докато системата не придобие в общи линии своя облик. След това следват преглед на всичко от страна на потребителя и писмено подаване на забележки и коментари към разработчика. След отстраняването на забележките се създава първия работещ вариант на приложението, който бива отново тестван от страна на потребителите. Следва зареждане на първоначалните данни и започва същинската работа със системата, като паралелно на това се правят прототипи на допълнително желаните функционалности. Продължава писменото подаване на забележки и отстраняването им. В момента, в които и двете страни са на мнение, че разработваната част от системата функционира нормално тя се интегрира с останалата част и така процеса продължава. Тук е важно да се отбележи, че след виждането на първоначалния прототип от страна на потребителя и отстраняването на подадените писмено забележки, всяка следваща промяна на мнението е важно да се договаря допълнително финансово. Така самият потребител внимава при приемането на системата и се чувства съпричастен със създаването й, и съответно процеса по внедряване върви по-гладко, а от друга страна разработчика няма чувството, че труда му се омаловажава. Опростен вариант на описаният модел е показан на диаграмата на фиг. 2.3.2.



Фиг. 2.3.2 Модел за разработка

3. Анализ и проект на разработката

3.1. Изисквания към системата

Както беше описано по-горе, целта на дипломната работа е създаването на система за водене на документооборота и склада на фирма за производство на електронни компоненти. В предишната точка беше обоснован избора на реализация с PHP и MySql, а тук ще бъдат разгледани по-подробно изискванията към приложението.

Целта на реализираната система е да се обхванат всички бизнес процеси на фирмата, да се облекчи правенето на справки, да се автоматизира процеса за внос и износ максимално. Системата трябва да е „лесно съпровождана" [2], „надеждна"[2], "ефективна и с удобен и лесен за усвояване потребителски интерфейс" [2]. Както и да е реализирана на достъпна цена [2].

Според цитирания източник за да се постигне качество трябва да се намери правилния синхрон между горните изисквания. Определенията за тях отново според същия източник [2] са:

За да бъде един софтуер лесно съпровождан, в него относително лесно трябва да могат да се правят изменения и подобрения, както и да бъде добре документиран .

Да е надежден означава, че грешките трябва да са сведени до минимум, а продукта да изпълнява очакваните потребителски функции.

Ефективен - да използва оптимално хардуера .

Това, даден софтуер да притежава удобен и лесен за усвояване потребителски интерфейс, става все по-важно и актуално, поради нарастването на броя на потребителите и разширяването на спектъра им. Ако даден продукт има много богата функционалност, но не добра разработен интерфейс, той рискува не само да се затрудни работата на потребителите и да ги настрои негативно, но дори те да не забележат съществуването на дадена функционалност и съответно да не я използват.

Основната дейност на фирмата е производството на медицински електронни апарати за износ. За целта тя получава елементите, от които се правят по-сложните изделия обикновено чрез внос от компанията , за която се изнасят готовите детайли. Този бизнес процес изглежда по следния начин:

Получава се поръчка за доставка на изделие в Боуви България. Следва подаване от страна на Боуви България към доставчика, от когото е дошла поръчката, на така наречената проформа-фактура, в която вече е специфицирана цената на това, което ще се достави. Остава да се попълнят сроковете за доставка на определените количества. Дадена проформа може да се разбие на много дати за доставки. След като се попълнят документите кога, какво и колко трябва да бъде изнесено се преминава към планиране на компонентите, от които ще бъде произведено изделието за износ. Установяват се наличностите и недостига се поръчва за доставка.

При внос на компоненти има два вида митнически декларации. Първия вид, през който минава по-голяма част от вноса са така наречените митнически декларации за активно усъвършенстване. При тях фирмата, която ще получи готовото изделие, изпраща на фирмата производител детайлите, от които това изделие ще бъде произведено. По този начин се избягват плащанията на мита. Особеното на метода е, че до една година след като елементите са внесени, те образно казано отново трябва да са напуснали страната вече под

формата на готовата продукция. Именно за това се поставя изискване елементи внесени на активно усъвършенстване да бъдат изнасяни с приоритет. Освен това непрекъснато трябва да се знае къде се намират елементите внесени по този начин. Например ако бъдат предадени на фирма подизпълнител, в приемо-предавателния протокол, трябва изрично да се каже от коя митническа декларация са тези елементи, за да може при проверка да бъде установено къде са те.

Другия тип митнически декларации са за редовен внос. Там митата се плащат и няма по-нататъшни изисквания за елементите от страна на Митническите власти.

Ако поръчката за доставка на материалите, които не достигат е към външна фирма то при пристигането им се прави една от обяснените по-горе митнически декларации за внос. Те завеждат материалите в склада. Ако поръчката е към вътрешен доставчик (например при закупуване от магазин), тогава при доставянето на елементите се въвежда складова разписка за тях. Тя е документа, който вкарва в склада доставените по поръчката елементи.

Трябва да има възможност и за бракуване на внесените по активно усъвършенстване елементи. Това става с митническа декларация за унищожение.

За окачествяване на произведените детайли се използва протокол за качество. Той се попълва независимо дали даден компонент е произведен от самата фирма или от външен подизпълнител.

Елементи се предават на подизпълнител, като се попълва приемо-предавателен протокол. Единственият случай, при който се предават елементи, е когато към дадения подизпълнител има поръчка за доставка на сложен компонент, съставните на който са предаваните с протокола. В приемо-предавателния протокол трябва да се пише поръчката към доставчика, а и след това при приемане на готовото изделие, при писането на складова разписка и протокола за качество, отново трябва да се напише по коя поръчка става това. По този начин се осигурява проследимост на компонентите и съответно винаги при откриване на дефект в някой детайл на готово изделие, може да се открие, от къде е дошъл всеки негов детайл до най-ниско равнище. Фирмата работи и с два вида фактури. Едните са за чужбина при заплащане на изнесена продукция, а другите българските им аналози.

Важна функционалност е улесняването и автоматизирането на процеса на поръчване на необходимото количества елементи, за да бъдат удовлетворени поръчките. Това става като се отчитат не само складовите наличности, но и вече поръчаните неща за доставка.

Достъпа до системата трябва да бъде ограничен посредством потребителски имена и пароли. Необходимо е и да има разделяна на правата за достъп на такива, които имат пълни права и такива, които имат само права за изпълнение на справки.

За да бъде реализирана цялата горепосочена функционалност трябва да се поддържа номенклатура от основни записи на компоненти и техните рецептурници, доставчици, потребители и др.

Поставено е и изискване към основния запис на компонент да може да се прикача допълнителна информация под формата на файл.

Предвидено е и използването на статични данни като валути, типове документи, мерни единици и др. За тяхната поддръжка, за сега не е необходимо да се създава потребителски интерфейс.

На последно място, но не и по важност е създаването на необходимия брой справки. В следващия списък те са изредени и описани накратко:

- Складови наличности

    - Движение на компонент – Да дава възможност при избор на елемент, интервал от време и тип на документ, да се види участието на избрания компонент в документите. Да сумира количествата по тип на документите и накрая общата сума за интервала. Ако са обхванати всички документи за този компонент крайната сума да представлява наличността на компонента в склада;
    - По днешна дата – Текущото състояние на склада по компоненти;
    - По фирми - в склада – При избор на доставчик, интервал от време и тип на документите да дава наличните от този доставчик елементи, които още не са изнесени;
- Митнически декларации
    - Всички – Да показва всички митнически декларации за внос, с техните позиции от елементи, и за всяка позиция списък с декларациите, с които са изнасяни количества от тази позиция;
    - По документи – Както горната, но с възможност за избор на декларация;
- Обща справка за документи – Селекция по всички възможни критерии, които притежава документ. Да показва всички характеристики на документа и неговите позиции в списъчен вид;
- Елементи BOM (bill of material) – В рецептурниците на кои компоненти фигурира избрания при селекцията компонент.
- Разходни норми BOM – За избран компонент да се показват съставните му. Да има възможност за разбивка до различни нива.
- Компоненти – Да показва митническото движение за избран компонент. С кои декларации е внасян и с кои изнасян;
- Разходна норма за митница – За избрана митническа декларация за износ да се показват материалите внесени за активно усъвършенстване, които тя сега експортира обратно;
- Планиране - Вземайки в предвид отворените проформи, отворените поръчки към доставчик и складовите наличност към избрана дата, да даде необходимите за поръчване компоненти и да облекчи създаването на поръчки като автоматично попълва повечето данни в тях.
- Проследимост – По избран протокол за качество да се генерира списък с произхода на съставните на окачествяваното изделие. При наличие на сложен компонент да може да се извика справката рекурсивно за него;
- Показване компонент – По избран компонент да показва характеристиките му, както и неговите съставни.
- Справка за проформа-фактура – Да има възможност да се прегледат проформите по доставчик или по номер на документ за даден интервал от време. Резултатът трябва да бъде в таблична форма, като под всяка позиция на проформа се изрежда списък с фактурите, които са изнасяли от тази позиция. Трябва ясно да се вижда и остатъка по позиции на проформата като количество и цена;
- Справка за поръчка за доставка – Същата като за проформа фактурата, но за поръчка за доставка и съответно складови разписки или митнически

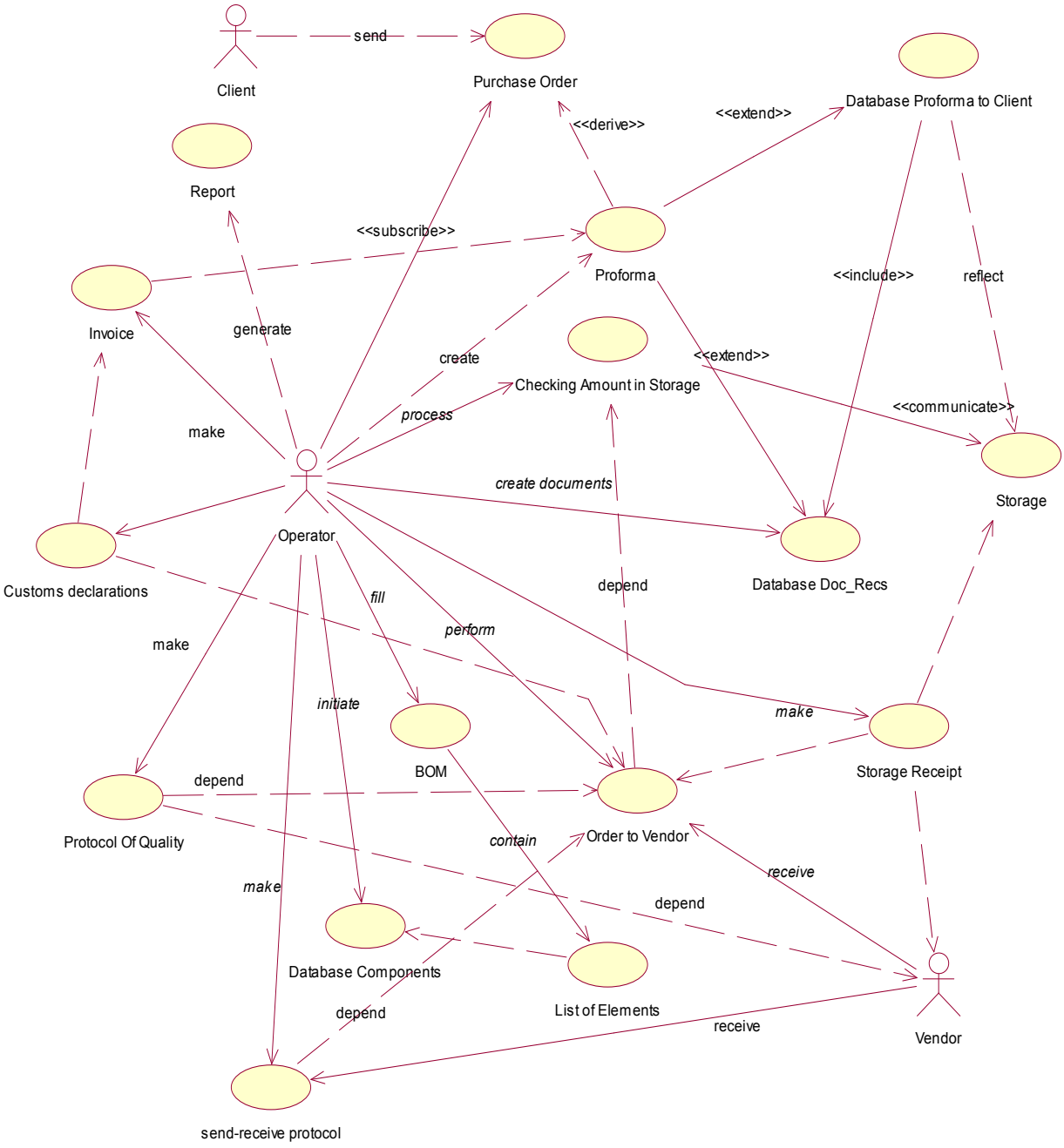
декларации, които променят остатъчните количества и цени на позициите, на поръчката за доставка.

За справките трябва да бъде предвидена и възможност за множествен избор на доставчик при необходимост.

3.2. Функционален модел

При планирането на реализацията е избрано представяне, което изисква минимални корекции при добавяне на нова функционалност, за да бъде решението по-гъвкаво и системните изисквания да са минимални.

На първо място по ред на проектиране стои диаграмата на потребителските случаи (Use Case Diagram) – фиг. 3.2.1 . На нея е показана същината на бизнес процесите като са спестени някои от по-маловажните и странични процеси, за опростяване на модела.



Фиг. 3.2.1 Диаграмата на потребителските случаи

Ролята на клиента се състои в това да подаде поръчка за доставка (Purchase order) както и да бъде адресант на митническите декларации за износ и на фактурите. В системата поръчката за доставка се отразява чрез попълване на форма, при създаването на проформа фактура от оператора.

Основните документи, които отразяват взаимоотношенията с доставчика са: поръчка за доставка, приемо-предавателен протокол, складова разписка и митническа декларация. Когато даден доставчик се явява подизпълнител на Боуви-България ЕООД, той първо получава поръчка за доставка на произвежданото от него изделие, следва приемо-предавателен протокол на частите, които се предават на доставчика (от тях ще се произведе готовото изделие). След произвеждането на продукта, той се приема със складова разписка и се окачествява, с протокол за качество или се бракува, с протокол за брак. От фигурата се вижда, че поръчката за доставка обвързва приемо-предавателния протокол, складова разписка и протокола за качество. По този начин се обхваща целия бизнес процес по изготвянето на изделие от подизпълнител.

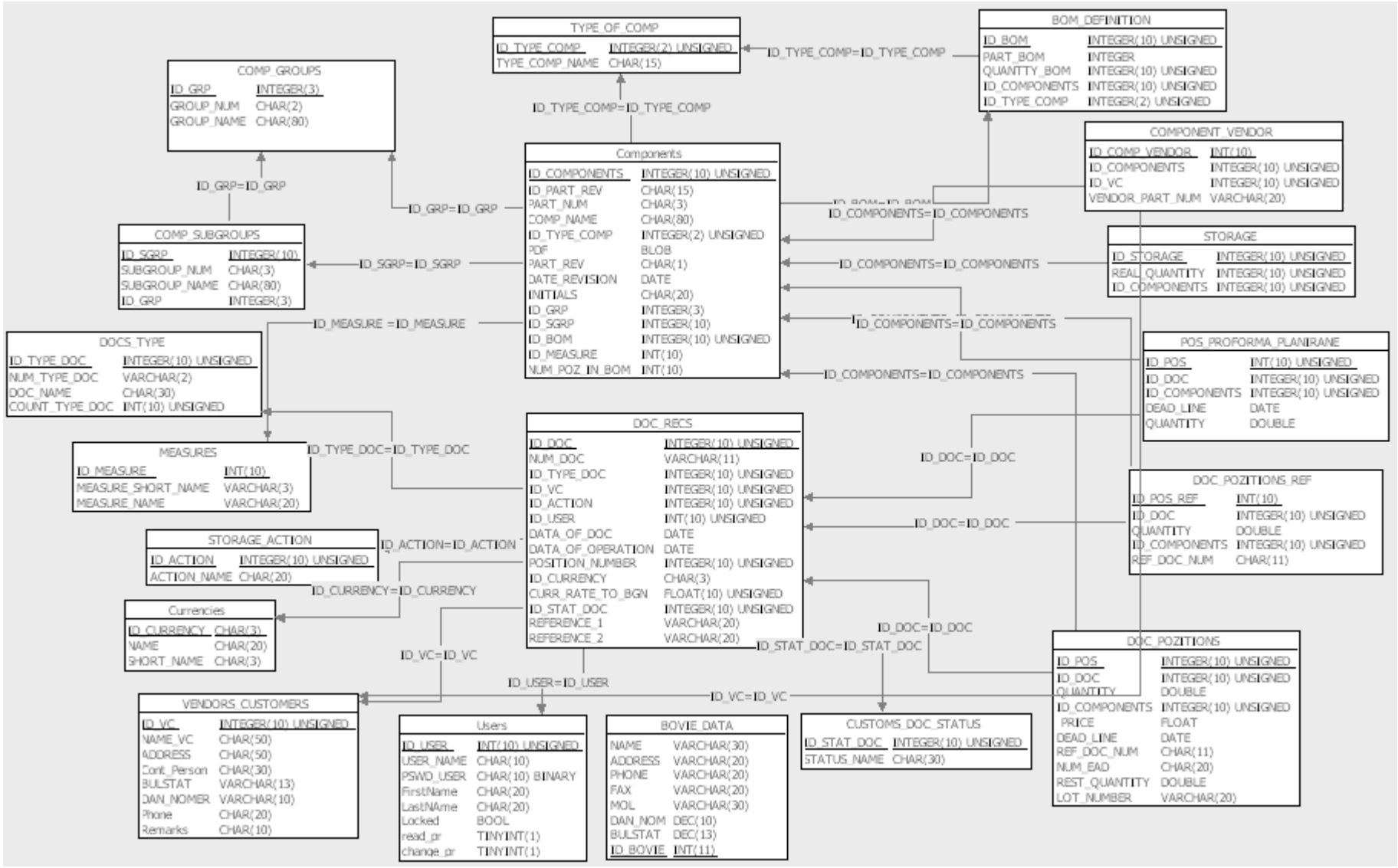
Оператора се грижи за поддържането на актуална номенклатура от компоненти, доставчици и т.н както и за изготвянето на всички документи.

Отношенията между документите, т.е. как създаването на един документ променя статуса на друг както и самите документи, по-подробно разгледани като същност и процеси, са обяснени в по-долните точки.

3.3. Модел на базата данни

Представен е *clay* модела на базата от данни. Виж фиг. 3.3.1. На него могат да се видят връзките между таблиците, за които са дефинирани и ограничителни условия. Основните таблици са *components, doc_recs, doc_pozitions*. Връзките на другите таблици към тях са осъществени чрез индекси като целта е да се избегне дублирането на данни и базата да е максимално нормализирана. Нормализирането на базата данни, не винаги е било водещо за избора на структурата й. Например таблицата *storage* съзнателно е изнесена като отделна, а не е дефинирана чрез допълнително поле в *components*, защото реално се използват много по-малко компоненти, отколкото са дефинираните в *components*. В *storage* се записват само тези елементи и материали, по които има движение чрез документи. Така се избягва обхождането на голяма таблица като *components* и се постига по-голямо бързодействие.

Всички таблици, с тяхната структура и предназначение, са обяснени в детайли в следващата глава.



Фиг. 3.3.1. Модел на базата данни

4. Софтуерна реализация

4.1. База от данни

4.1.1 Общи положения

Базата данни е изградена като InnoDB тип. Таблиците са свързани с така наречените ограничения. Така самата база се грижи за това да не позволява създаването на неконсистентност в нея като бъдат изтрити вече използвани индекси или като се направят нови записи с невалидни ключове. Това улеснява процеса на програмиране като спестява много проверки. Друг голям плюс на типа InnoDB е че поддържа използването на транзакции. Навсякъде където има повече от един запис в таблица, за да е успешно дадено действие, се използват транзакции. Така се гарантира извършването на операцията до край (commit) или нейното връщане до изходно състояние (rollback).

4.1.2 bom_definition

Поле	Тип	Описание
BOM_INDEX	int(10)	Primary key, Auto increment
ID_BOM	int(10)	връзка с components, той обединява BOM на компонента
PART_BOM	bigint(10)	Пореден номер за BOM-а
QUANTTY_BOM	double	Количество
ID_COMPONENTS	int(10)	Идентификатор на компонент от components. Индексно поле.

В нея се съхраняват данните за съставните, на един компонент, елементи (BOM – bill of material)

4.1.3 bovie_data

Поле	Тип	Описание
NAME	varchar(30)	Наименование
ADDRESS	varchar(55)	Адрес
PHONE	varchar(20)	Телефон
FAX	varchar(20)	Факс
MOL	varchar(30)	Материално отговорно лице
DAN_NOM	decimal(10,0)	Данъчен номер
BULSTAT	decimal(13,0)	Булстат
ID_BOVIE	int(11)	Primary key, Auto increment

Данни за фирмата. Използва се за да се избегне данните да са вградени в HTML кода, за по-лесна промяна. Няма индекси или връзки с други таблици.

4.1.4 comp_groups

Поле	Тип	Описание
ID_GRP	int(3)	Primary key, Auto increment
GROUP_NUM	char(2)	Уникално поле за номер на групата
GROUP_NAME	char(80)	Описание на групата

Таблица за групите на компонентите.

4.1.5 comp_subgroups

Поле	Тип	Описание
ID_SGRP	int(10)	Primary key, Auto increment
SUBGROUP_NUM	char(3)	Номер на подгрупата
SUBGROUP_NAME	char(80)	Описание на подгрупата
ID_GRP	int(3)	Индекс за връзка с comp_groups

Таблица за подгрупите на компонентите. Комбинацията от SUBGROUP_NUM и ID_GRP е уникална.

4.1.6 component_vendor

Поле	Тип	Описание
ID_COMP_VENDOR	int(10)	Primary key, Auto increment
ID_COMPONENTS	int(10)	Индекс за връзка с components (компоненти)
ID_VC	int(10)	Индекс за връзка с vendors_customers (клиенти/доставчици)
VENDOR_PART_NUM	varchar(20)	Номер на компонент на доставчика ако е различен от този на Боуви

Таблицата се използва за специфициране на номера на доставчици, които имат по-различна номенклатура от тази на Боуви. Попълва се при създаване на поръчка еднократно или при промяна на номерата за доставчик.

4.1.7 components

Поле	Тип	Описание
ID_COMPONENTS	int(10)	Primary key, Auto increment
ID_PART_REV	char(15)	Уникален номер на компонент. Сборен от група, подгрупа, парт номер и ревизия
PART_NUM	char(3)	Номер на част (part number)
COMP_NAME	char(80)	Наименование
ID_TYPE_COMP	int(2)	Индекс за връзка с type_of_comp (определя типа на компонента)
ID_MEASURE	int(3)	Индекс за връзка с measures (определя мерната му единица)
PDF	blob	Поле за съхранение на път за файл
PART_REV	char(1)	Ревизия
DATE_REVISION	date	Дата на ревизията
INITIALS	char(20)	Инициали за ревизията
ID_GRP	int(3)	Индекс за връзка с comp_groups (определя групата на компонента)
ID_SGRP	int(10)	Индекс за връзка с comp_subgroups (определя подгрупата на компонента)
ID_BOM	int(10)	Номер идентифицираш съставните на този компонент. Връзката му с bom_definition
NUM_POZ_IN_BOM	int(10)	Колко позиции има в bom_definition за този компонент, т.е. колко са неговите съставни на първо ниво.

Таблица за компонентите. Тя е най-важната таблица от тези за основни записи на компонентите. В нея се съхраняват всички компоненти, а множеството от връзки към другите таблици, нормализират базата и дават техните характеристики.

4.1.8 currencies

Поле	Тип	Описание
ID_CURRENCY	char(3)	Primary key, Auto increment
NAME	char(20)	Наименование
SHORT_NAME	char(3)	Кратко име. Уникално поле.

Таблица за валутите. Статична таблица. Предвидено е да се променя директно през DB администратора.

4.1.9 customs_doc_status

Поле	Тип	Описание
ID_STAT_DOC	int(10)	Primary key, Auto increment
STATUS_NAME	char(30)	Описание на статуса

Таблица за статусите на документите. Статична таблица. Предвидено е да се променя директно през DB администратора.

4.1.10 doc_pozitions

Поле	Тип	Описание
ID_POS	int(10)	Primary key, Auto increment
ID_DOC	int(10)	Индекс за връзка с doc_recs (определя за кои документ е тази позиция)
QUANTITY	double	Количество
ID_COMPONENTS	int(10)	Индекс за връзка с components (определя компонента, които е на тази позиция)
PRICE	float	Цена
DEAD_LINE	date	Краен Срок за доставка.
REF_DOC_NUM	char(11)	Референтен номер – в различните типове документи реферира към различни типове. Например при митническата декларация е за номер на поръчка. При фактурите е номер на проформа.
NUM_EAD	char(20)	Номер на митническа декларация с която е внесе компонента (не за всички документи се използва)
REST_QUANTITY	double	Остатъчно количество за компонента по позицията (не за всички документи се използва)
LOT_NUMBER	varchar(20)	Лот номер на компонента

Таблица, в която се записват данните за позициите на документа. Връзката с основния запис на документа е ID_DOC.

4.1.11 doc_pozitions_ref

Поле	Тип	Описание
ID_POS_REF	int(10)	Primary key, Auto increment
ID_DOC	int(10)	Индекс за връзка с doc_recs (определя за кои документ е тази позиция)
QUANTITY	double	Количество
ID_COMPONENTS	int(10)	Индекс за връзка с components (определя компонента, които е за тази позиция)
REF_DOC_NUM	char(11)	Референтен номер – в различните типове документи реферира към различни типове

Таблица, в която се записват референтни документи, за някои документи. Например при митническите декларации за износ се пише от кои декларации е внасян изнасяния компонент.

4.1.12 doc_recs

Поле	Тип	Описание
ID_DOC	int(10)	Primary key, Auto increment
NUM_DOC	varchar(11)	Номер на документ. Уникално поле.
ID_TYPE_DOC	int(10)	Индекс за връзка с type_of_comp (определя типа на компонента)
ID_VC	int(10)	Индекс за връзка с vendors_customers (клиенти/доставчици)
ID_ACTION	int(10)	Индекс за връзка със storage_action
ID_USER	int(10)	Индекс за връзка с users (кои е създал документа)
DATA_OF_DOC	date	Дата на документа
DATA_OF_OPERATION	date	На коя дата е създаден документа
POSITION_NUMBER	int(10)	Брой на позициите в него
ID_CURRENCY	char(3)	Индекс за връзка със currencies (тип на валута)
CURR_RATE_TO_BGN	float	Курс на валутата към лева
ID_STAT_DOC	int(10)	Индекс за връзка със customs_doc_status (статус на документа)
REFERENCE_1	varchar(20)	Допълнително поле 1
REFERENCE_2	varchar(20)	Допълнително поле 2

Таблицата, в която се пишат основните данни за документа. Номерът на документа е 10 цифрен номер, който се определя по следния начин:

Първите 2 цифри се взимат в зависимост от типа на документа, а последните 8, в зависимост от последния номер на документ от този тип. Текущия номер на документите от различните типове се пази в docs_type.

4.1.13 docs_type

Поле	Тип	Описание
ID_TYPE_DOC	int(10)	Primary key, Auto increment
NUM_TYPE_DOC	varchar(2)	Двуцифрен номер идентифициращ типът. Уникално поле.
DOC_NAME	char(30)	Наименование на типа документ
COUNT_TYPE_DOC	int(10)	Текущ последен номер на документ от този тип

Таблица за видовете документи. Използва се и за определяне на номер на документ при създаване на нов в зависимост от полето COUNT_TYPE_DOC. Статична таблица. Ако се налага да се добавят нови типове документи това трябва да стане през DB администратора. Последните текущи номера се обновяват от функцията CreateDocument при успешно създаване на документ.

4.1.14 measures

Поле	Тип	Описание
ID_MEASURE	int(3)	Primary key, Auto increment
MEASURE_SHORT_NAME	varchar(3)	Кратко име на мерната единица. Уникално поле.
MEASURE_NAME	varchar(20)	Наименование на мерната единица

Таблица за мерните единици. Статична таблица. Предвидено е да се променя директно през DB администратора.

4.1.15 pos_proforma_planirane

Поле	Тип	Описание
ID_POS	int(10)	Primary key, Auto increment
ID_DOC	int(10)	Индекс за връзка с doc_recs (определя за кои документ е тази позиция)
ID_COMPONENTS	int(10)	Индекс за връзка с components (определя компонента, които е за тази позиция)
DEAD_LINE	date	Краен срок за доставка
QUANTITY	double	Количество

Таблица свързана с функциите по планирането. В нея по-време на създаването на проформа фактура се попълват сроковете за доставка на определени количества от компонентите за тази проформа. Например трябва да се доставят по проформа 100 апарата, но доставката трябва да бъде разбита на 10 пъти по 10 апарата. Това разбиване се запазва в тази таблица.

4.1.16 storage

Поле	Тип	Описание
ID_STORAGE	int(10)	Primary key, Auto increment
REAL_QUANTITY	double	Количество в склада
ID_COMPONENTS	int(10)	Индекс за връзка с components (определя компонента)

Таблица за текущото състояние на склада. В нея се отразяват всички промени, които настъпват при създаването или промяната на документ и тя е актуалното дава текущо състояния на склада по документи.

4.1.17 storage_action

Поле	Тип	Описание
ID_ACTION	int(2)	Primary key, Auto increment
ACTION_NAME	char(20)	Описание на действието за склада

Таблица за начина на действие на документ към склада. Статична таблица. Предвидено е да се променя директно през DB администратора. Текущо не се използва реално.

4.1.18 temp_bom_definition

Поле	Тип	Описание
ID_BOM	int(10)	Идентификатор за BOM на компонент
PART_BOM	int(5)	Номер на компонента в BOM
QUANTTY_BOM	int(10)	Количество
ID_COMPONENTS	int(10)	Индекс за връзка с components (определя компонента)
ID_TYPE_COMP	int(2)	Индекс за връзка с type_of_comp (определя типа на компонента)

Работна таблица за временно съхранение на BOM на компонента при създаването на нов такъв. При редактирането на съществуващ или при създаването на нов с копиране от стар, тя не се използва заради специфика на интерфейса.

4.1.19 type_of_comp

Поле	Тип	Описание
ID_TYPE_COMP	int(2)	Primary key, Auto increment
TYPE_COMP_NAME	char(15)	Наименование на типът компонент

Таблица за видовете компоненти. Статична таблица. Предвидено е да се променя директно през DB администратора.

4.1.20 users

Поле	Тип	Описание
ID_USER	int(10)	Primary key, Auto increment
USER_NAME	char(10)	Потребителско име
PSWD_USER	char(50)	Парола
FirstName	char(20)	Първо име
LastNAme	char(20)	Фамилно име
Locked	tinyint(1)	Статус на потребителя
read_pr	tinyint(1)	Наличност на права за четене
change_pr	tinyint(1)	Наличност на права за промяна

Таблица за потребителите на системата. Само един специален потребител, може да оперира с други. Това е потребителя bovie. Другото особено за таблицата е, че паролата е кодирана като MD5.

4.1.21 vendors_customers

Поле	Тип	Описание
ID_VC	int(10)	Primary key, Auto increment
NAME_VC	char(50)	Наименование
ADDRESS	char(50)	Адрес
CONT_PERSON	char(30)	Лице за контакти

BULSTAT	varchar(13)	Булстат номер
DAN_NOMER	varchar(10)	Данъчен номер
PHONE	char(20)	Телефон
REMARKS	char(100)	Бележки

Таблица за клиенти, доставчици. Като такива се представят и отделните производства на Боуви, за да може да се следят като отделни части от централата и за да е ясно в кой момент какви поръчки изпълняват и какви компоненти имат предадени за работа

4.2. PHP реализация

От гледна точка на програмиста решението може да се разглежда като разделено на три части:

1. Обща част, която се състои от обединяващите за цялата система визуални елементи, както и общи функции, които осигуряват валидацията на сесиите, на съответните права за достъп и др. Те се включват в началото на PHP кода посредством оператора *require.* Менютата са създадени с помощта на Xara WebStyle 4.0 и няма да бъдат разглеждани в тази точка;
2. Специфична част, характеризираща особеностите на съответната функционалност, в която чрез скриптове е реализирана тази функционалност. В тази точка ще бъде по-подробно разгледана именно те;
3. Изградената база от данни, която със своите таблици, индекси, връзки и ограничения е важна и неделима част от разработката.

От потребителска гледна точка системата може да се раздели на :

1. Функционалност за управление на основните записи в системата (компоненти, доставчици, потребители и т.н.);
2. Функционалност за създаване на документооборота;
3. Справки.

4.2.1. Елементи

Групи компоненти

Всеки компонент принадлежи към дадена група компоненти. Групата на компонента се определя от двуцифрен уникален номер и име. Този двуцифрен уникален номер е поредица от два символа и участва в номера на всеки компонент, който принадлежи към дадената група. Данните за всяка отделна група компоненти се съхраняват в таблицата ‘*comp_groups’*.

Референтни файлове:

- Дефиниране

Define_groups.php -> Group_save.php

- Редактиране

Edit_groups.php -> Group_edit.php

- Изтриване

Delete_groups.php -> Group_del.php

Подгрупи компоненти

Всяка група компоненти може да бъде разделена на няколко подгрупи. Всяка подгрупа, на дадена група, се определя от трицифрен уникален номер в групата и име. Всеки компонент принадлежи към дадена подгрупа, на някоя група компоненти. Този трицифрен уникален номер е поредица от три символа и участва в номера на всеки компонент, който принадлежи към дадената подгрупа, на групата компоненти. Данните за всяка отделна подгрупа компоненти се съхраняват в таблицата '*comp_subgroups'*.

Референтни файлове:

- Дефиниране

Define_subgroups.php -> Subgroup_save.php

- Редактиране

Edit_subgroups.php -> Subgroup_edit1step.php -> Subgroup_edit2step.php -> Subgroup_edit3step.php

- Изтриване

Delete_subgroups.php -> Subgroup_del1step.php -> Subgroup_del2step.php

Компоненти

Всеки компонент принадлежи към дадена подгрупа, на дадена група. Компонентите се унифицират с т.н. номенклатурен номер. Този номенклатурен номер е уникален за всеки компонент в системата и се конструира от номера на групата, към която принадлежи компонента, номера на подгрупата му в тази група, номер на съответната част и ревизия на компонента (единствената незадължителна съставна част от номера). Номерът на частта е трицифрен идентификатор, а ревизията – един произволен символ. Комбинацията от номер на частта и ревизия е уникална за компонента от съответната подгрупа на съответната група. Примерно, компонент с номенклатурен номер 01-001-002 принадлежи към подгрупата от компоненти с номер 001 на група номер 01. Номерът на частта на този компонент е 002. Друг пример е номенклатурният номер 01-001-002-Q. Компонент с такъв номенклатурен номер принадлежи към подгрупата от компоненти с номер 001 на групата с номер 01 и има номер на частта 002 с ревизия Q.

Основния запис на компонент се съхранява в таблица *'components'*. Структурата и е разгледана най-подробно в точката база от данни. Това от какви елементи се състои даден компонент се съхранява в таблица *'bom_definition'*. Връзката между двете таблици е двустранна. Веднъж по полето ID_BOM от основния запис на компонент се определят всички позиции, които участват в BOM на този компонент. Всички позиции имат еднакъв ID_BOM. От своя страна те се характеризират с индекс ID_COMPONENTS към 'components'.

При дефиниране на компонент поради спецификата на това, че в Comp_save.php (виж референтни файлове) се прави дефинирането на BOM (рецептурник – Bill of Material) както и самото запазване на компонента в таблиците, като при дефиниране на BOM се дава възможност за връщане и за редактиране основните данни, се налага вече дефинирания BOM, да се съхранява във временна таблица. Тя се казва *'temp_bom_definition'*.

Отново при дефинирането на компонент, не се допуска да се дефинира BOM на компонент, който е от тип '*simple'* чрез техника на деактивиране или активиране на бутона, чрез който става дефинирането на BOM. Това е реализирано с JavaScript функция.

Началните екрани, с изключение на този при дефиниране, предлагат една и съща функционалност за избор на компонент и тя е:

Изборно поле, данните за което се извличат от таблицата *'components'*, с техните номера и имена. Дадена е възможност в поле за редактиране да се напише номера на желания компонент или чрез натискане на клавиша 'f', при активен елемент за избор на компонент, да се отвори помощна форма, с която да се улесни избора на компонент. В точката приложение е показана функцията FilterComponents(), с която се реализира използването на клавиша 'f'.

В полето, където може да се напише номера на компонента може да се използва и знака '%', за указване на множествена селекция. При заявката към базата данни това се реализира с оператора LIKE.

Описание на специфичните за всяка функция неща:

Дефиниране:

В началния екран за избор се избира групата, от която ще е дефинирания компонент. Това става посредством изборно поле, данните за поето са заредени от '*comp_group'*. При натискане на бутона за продължение се визуализира формата, където се попълват основните данни за компонента. От '*comp_subgroups, measures_ type_of_comp'* са заредени изборните полета съответно за подгрупа, мерна единица и тип на компонента. Останалите характеристики се попълват посредством текстови полета. Чрез JavaScript функции се следи данните да са коректни и всички задължителни полета да са попълнени. При натискане на бутона за дефиниране на BOM се преминава към Comp_save.php. Същия файл обработва и бутона "ЗАПИШИ". Това става чрез проверка в началото на файла кой бутон е натиснат и съответно разклоняване на алгоритъма.

```
if ($submit_button == 'Save') {
……..
} else {
…....
}
```

Друго особено нещо е дадената възможност за записване на файл към основния запис на компонент. Това обуславя html формата в *Comp_def2step.php* да е *"multipart/form-data"*. В таблицата се пази само пътя до файла, който се записва в директория '*files'*. Ограничението за големината на позволените файлове е настройка на Apache сървъра. Програмният код, който реализира качването на файла на сървъра е показан в приложенията.

При запис на компонента логическите проверки за това дали даден компонент се дефинира правилно са оставени на базата данни, която поради начина по който е конфигурирана, не позволява създаването на компонент с еднакъв номер, не позволява сбъркани индекси и т.н. Грешките, които генерира се прехващат и прекъсват изпълнението на алгоритъма след визуализирането на съобщение на екрана.

Редактиране:

При редактирането единственото особено нещо, е че не се позволява промяна на типа на компонента. Това спестява алгоритмична сложност и поради факта, че може никога да не се наложи да бъде правено, е прието това ограничение.

Изтриване:

След като бъде избран компонента за изтриване и се натисне бутона за изтриване, се визуализира екран за потвърждение. Дори да бъде потвърдено изтриването, то няма да бъде извършено ако този компонент вече е участвал в документи. Т.е. ако вече с него има някаква активност или ако е дефиниран друг компонент, в BOM-а на който изтривания компонент участва.

Копиране:

Служи за създаване на компонент, чрез копиране от друг компонент. За разлика от при дефиниране, тук създаването на BOM-а е на същия екран, което не налага използването на временна таблица за съхранението му. Типът му отново не може да бъде променян поради гореспоменатите причини.

Референтни файлове:

§ Дефиниране
Define_comp.php -> Comp_def2step.php -> Comp_save.php

§ Редактиране
Edit_comp_BOM.php -> Edit_comp_BOM_step2.php -> Save_edit_comp_BOM.php

§ Изтриване
Delete_comp.php -> Delete_comp_save.php

§ Копиране
Copy_BOM.php -> Copy_BOM_step2.php -> Save_copy_comp.php

§ Показване
Описано е в справки

4.2.2. Документи

Общи положения

Всеки документ съдържа няколко задължителни за попълване и няколко опционални полета. Сред основните данни за документа са датата на документа, доставчика/клиента, валутата на документа, курса на валутата спрямо BGN и две други полета, които в различните документи служат за различни референции. От друга страна всеки документ притежава уникален номер, тип, потребител, дата на операцията по създаването или модификацията му и евентуално статус. Тези основни данни за документа се записват в таблицата *‘doc_recs’*.

Уникалният номер на документа в общия случай е 10-цифрен номер, който се получава по следният начин: първите две цифри са номера на типа на документа, а

останалите 8 цифри са поредният номер на документа спрямо останалите документи в системата от същият този тип. Пример: 0100000020 е номер на документ от тип 01 (Митническа декларация за АУ) и до неговото създаване е имало създадени точно 19 други митнически декларации за АУ. Информацията за номер на типа и името на документите, както и за поредният номер за генериране на документ от съответният тип, се съхранява в таблицата *'docs_type'*.

Изключение от описаното правило за генериране на номер на документ правят американските фактури. Те получават номера си от номера на Проформа-Фактурата, към която реферират, плюс поредна буква от латинската азбука за поредната фактура, свързана със съответната Проформа-Фактура. Пример: 0600000020A е номер на фактура, свързана с проформа-фактурата 0600000020,както и фактурата 0600000020B.

Всеки документ притежава поле за статус, но не за всеки документ този статус носи смисъл. Документите, които се разглеждат като документи със статус са или отворени или затворени и реално само техният статус се изследва. Данните, свързани със смисъла на статуса на документите, се съхраняват в таблицата *'custom_doc_status'*.

Освен основни данни всеки документ може да има и позиции. Всяка позиция е свързана с компонент, количество от този компонент, евентуално цена, краен срок, номер на референтен документ към тази позиция, номер на декларация за внос, свързана с тази позиция, оставащо количество по позиция и LOT номер. Не всички полета са задължителни и/или необходими за различните типове документи. От друга страна някои документи имат задължително точно една позиция (примерно протокола за качество и протокола за брак). Данните за отделните позиции на документите се записват в таблицата *'doc_pozitions'*.

При всяко въвеждане на документ чрез JavaScript се правят проверки дали всички задължителни полета са попълнени и дали стойностите са коректни. Също така при изпълняването на заявка към базата данни, резултата винаги се прихваща и при грешка се визуализира съобщение на екрана и изпълнението на програмата се прекъсва.

При работа с последователност от зависими едно от друго обръщения към базата данни се използват транзакции, с които се гарантира, че при грешка базата данни ще бъде възстановена в изходно състояния и ще остане консистентна. Реализацията на гореспоменатите проверки и пример за работа с транзакции са показани в приложенията.

Митническа декларация за АУ

Митническите декларации за АУ са документи, чрез които се зачисляват в склада материалите внесени през митницата, които са за активно усъвършенстване. По този начин таблицата *'storage'* се обновява с доставените чрез декларацията количества.

Всеки такъв документ съдържа няколко задължителни за попълване и няколко опционални полета. При избор на доставчик в полетата за поръчка от позициите на документа се зареждат всички отворени поръчки към избрания доставчик. В случай, че такава поръчка е избрана, тази декларация за АУ участва при затварянето на избраната поръчка. Това буквално значи, че всички декларации за АУ, декларации за редовен внос и складови разписки, които са посочили, че доставят материали за тази поръчка, ще участват сумарно в нейното затваряне. Ако доставените количествата по дадена позиция от поръчката са равни или надвишават поръчаните, то количествата от тази позиция се считат за изцяло доставени и излишествата се игнорират. Ако всички количества от всички

позиции на поръчката са доставени изцяло, то тази поръчка променя статуса си на затворена.

В митническата декларация за АУ е възможно да бъдат въведени две позиции с еднакви данни, както и позиции с еднакви компоненти, но специфични цени, количества и/или LOT номера. Изчерпването на позициите в митническите декларации за АУ става по номер на позиция в документа, т.е. по-предна позиция в документа гарантира по-ранно изчерпване на материалите от тази позиция.

Една особеност на митническите декларации за АУ е, че тя участва в различните справки чрез въведения за нея номер от митница. Ако такъв номер не е въведен, то тогава и само тогава се показва нейния 10-цифрен номер, с който тя реално се идентифицира в системата.

Друга особеност на митническите декларации за АУ е възможността за изкуственото им блокиране. Когато една декларация е блокирана, изчерпването на остатъчните количества по нея е замразено и системата я третира като затворена (примерно до разблокирането и).

Функциите, които се използват при записа на документа в базата са: **CreateDocument, InputInStorage, ManageDocStatusPor.** Те се използват при създаването не само на този документ и поради тяхната важност кодът им е показан в приложенията.

Референтни файлове:

- Създаване

Cust_AU_step0.php -> Cust_AU_step1.php -> Cust_AU_save.php

- Редактиране

Edit_Cust_AU_step0.php -> Edit_Cust_AU_step1.php -> Edit_Cust_AU_save.php

- Блокиране

Block_Cust_AU_step0.php -> Block_Cust_AU_step1.php -> Block_Cust_AU_save.php

Митническа декларация за редовен внос

Митническите декларации за редовен внос са документи, чрез които се зачисляват в склада материалите внесени през митницата, които са редовен внос. По този начин таблицата *'storage'* се обновява с доставените чрез декларацията количества.

Всеки такъв документ съдържа няколко задължителни за попълване и няколко опционални полета. При избор на доставчик в полетата за поръчка от позициите на документа се зареждат всички отворени поръчки към избрания доставчик. В случай, че такава поръчка е избрана, тази декларация за редовен внос участва при затварянето на избраната поръчка. Това буквално значи, че всички декларации за АУ, декларации за редовен внос и складови разписки, които са посочили, че доставят материали за тази поръчка, ще участват сумарно в нейното затваряне. Ако доставените количествата по дадена позиция от поръчката са равни или надвишават поръчаните, то количествата от тази позиция се считат за изцяло доставени и излишествата се игнорират. Ако всички количества от всички позиции на поръчката са доставени изцяло, то тази поръчка променя статуса си на затворена.

В митническата декларация за редовен внос както при тази за активно усъвършенстване е възможно да се въвеждат различни позиции за еднакви компоненти.

Една особеност на митническите декларации за редовен внос е, че тя участва в различните справки чрез въведения за нея номер от митница. Ако такъв номер не е въведен, то тогава и само тогава се показва нейния 10-цифрен номер, с който тя реално се идентифицира в системата.

Референтни файлове:

- Създаване

Cust_import_step0.php -> Cust_import_step1.php -> Cust_import_save.php

- Редактиране

Edit_Cust_import_step0.php -> Edit Edit_Cust_import_step1.php -> Edit_Cust_import_save.php

Митническа декларация за износ

Митническите декларации за износ са документи, чрез които се изнасят компоненти от склада през митницата. По този начин таблицата *'storage'* се обновява с изнесените чрез декларацията количества. Функцията, която прави това е OutputFromStorage (виж приложенията за дефиницията на функцията). Тези материали в общия случай са произведени от „Боуви-България" ЕООД или фирма-посредник, използвайки материали внесени на АУ или редовен внос. Когато при износа на даден компонент през митницата в производството му е участвал поне един компонент, внесен на АУ, тогава митническата декларация е декларация за реекспорт. Ако при производството на изнасяният компонент не участва нито един компонент, внесен на АУ, то декларацията е за редовен износ.

Всяка декларация за износ, съдържа няколко задължителни за попълване и няколко опционални полета. При избор на получател в полетата за фактура (Invoice) от позициите на документа, се зареждат всички фактури към избрания получател. В полето където се въвежда номер на компонент, може да се напише директно неговия номер или посредством бутона за избор да се избере такъв. При натискане на бутона за избор е дадена възможност да се използва помощната функция 'f' , за улесняване на търсенето. Посредством JavaScript функции се правят проверки за коректността на входните данни. След натискането на бутона за продължение се проверява за всеки елемент дали е разложим и ако е, тогава се проверява дали има внесени такива с декларация за АУ или за редовен внос. Ако елемента е внесен с декларация за АУ тогава се маркира като такъв, който не трябва да се разлага, а ако има внесен с декларация за редовен внос тогава се задава въпрос дали да се продължи с разлагането му или да се вземе този от декларацията за редовен внос. Проверките стават като в единия случай се проверява в *doc_pozitions*, *doc_recs*, дали има митническа за АУ, с остатък по позиция на същия компонент, по-голям от нула и дата на документа по-малка или равана от датата на митническата за износ. Проверката за това дали няма такъв компонент, но за редовен внос е същата но се проверяват митническите за редовен внос. Визуализират се списък с компоненти, за които трябва да се предприемат следните действия:

Да се изберат компонентите, които да *не* бъдат разлагани до по-прости при износ.

Предложените количества са максималните, които могат да бъдат взети при

текущите декларации за АУ и редовен внос. Задължително избраните компоненти са тези, по които има внос на АУ.

След като отново бъде натиснат бутона за продължение се пристъпва към самия запис на документа. Прави с проверка дали наличностите в склада са достатъчни. Това става чрез функцията OutputFromStorage(); Ако не, са се визуализира съобщение и документ не се създава. Извличат се данните за съставните на компонента – функция GetComponentPart(). Дефиницията и може да бъде видяна в приложенията. Обработват се масивите като вече е избрано кои елементи да бъдат разлагани докрай и кои не. Прави се и проверка дали има внесени до момента на износа достатъчно компоненти. Това се прави защото е възможно декларацията за износ да е в миналото и макар че в склада в момента има достатъчно количества за износ, за датата в която е самия износ не е ясно това дали е така само чрез разглеждането на таблица storage. Ако и тази проверка е задоволена се преминава към проверка дали има митнически декларации за внос, остатъците на които вече са станали нула (след износа). Ако има такива статуса им се променя на затворени.

Създава се самия документ с функцията CreateDocument(); и след това се запълва и референтната таблица (doc_pozitions_ref) с функцията CreateRefPositions() (виж приложенията за дефинициите на функциите). Там се записват от кои декларации за внос е черпено при износа. Всички тези записи в базата данни са сложени в транзакция, която се потвърждава (COMMIT) чак след като всички записи са минали успешно. Ако по време на транзакцията възникне грешка, то базата се връща в изходно състояние (ROLLBACK). По този начин се гарантира запазването на консистентността на базата данни.

Затварянето на митническите декларации за внос става по позиции, т.е. по номер на позиция – по преден номер се затваря по-рано.

При редактирането логиката е същата както при създаването, само че преди да бъдат правени проверките за наличностите в склада и по декларации, се възстановяват остатъчните количества по входните документи и се възстановява склада както е бил преди износа.

Референтни файлове:

- Създаване

Cust_AllE_step0.php -> Cust_AllE_step1.php -> Cust_AllE_save0.php -> Cust_AllE_save.php

- Редактиране

Edit_Cust_AllE_step1.php -> Edit_Cust_AllE_step2.php -> Edit_cust_E_save0.php -> Edit_cust_E_save.php

Митническа декларация за унищожение

Митническите декларации за унищожение представляват документ, които унищожават компоненти внесени по АУ.

В екрана за въвеждането им има само възможност за избор на компонент и на какво количество от него ще се унищожава. Както и текстово поле за дата на документа и за митническа декларация за внос.

След натискане на бутона за продължение се преминава през същия алгоритъм както при износ.

Прави с проверка дали наличностите в склада са достатъчни. Това става чрез функцията *OutputFromStorage();* Ако не са, се визуализира съобщение и документ не се създава. Прави се и проверка дали има внесени до момента на износа достатъчно компоненти. Това се прави, защото е възможно декларацията да е в миналото и макар че в склада в момента има достатъчно количества за износ, за датата в която е самото унищожение не е ясно това дали е така само чрез разглеждането на таблица *storage*. Ако и тази проверка е задоволена се преминава към проверка дали има митнически декларации за внос, остатъците на които вече са станали нула (след износа). Ако има такива статуса им се променя на затворени.

Създава се самия документ с функцията CreateDocument(); и след това се запълва и референтната таблица (*doc_pozitions_ref*) с функцията CreateRefPositions(). Там се записват от кои декларации за внос е черпено при износа.

Затварянето на митническите декларации за внос става по позиции, т.е. по номер на позиция – по преден номер се затваря по-рано.

Референтни файлове:

- Създаване

Cust_D_step0.php -> Cust_D_stepя1.php -> Cust_D_save.php

- Редактиране

Edit_Cust_D_step1.php -> Edit_Cust_D_step2.php -> Edit_cust_D_save.php

Проформа-Фактура

Проформата е документа, чрез който се определя на кого, какво трябва да бъде доставено, кога и на каква цена. Посредством частта от нея за разбиването на сроковете за доставка тя играе основно роля в планирането. В главния екран се попълват дата на документа, има полета за избор на клиент, данните за което са извлечени от *vendors_customers*, компонент, въвеждането на който може да стане или в текстовото поле или посредством бутона за избор, който визуализира стандартната за системата форма за избор на компонент с активна функция за улесняване на търсенето (‘f’), полета за количество и цена и срок на документа. Има и два бутона за отметка, които специфицират дали в изходния документ да има печат и дали да се премине през екран за планиране. При натиснат бутон за преминаване през планиране се визуализира форма, в която потребителя има възможност да разбива доставяния компонент на различни дати за доставка. За всяка дата има и количество. В резултат се попълва таблица *pos_proforma_planirane*, която се използва за планирането на доставяните материали. При натискане на бутона за продължение, в зависимост от това дали са попълнени дати за доставки, за цялото количество по проформата или не, се визуализират различни съобщения като единственото, при което не се допуска продължение е това, че са планирани за доставка по-големи количества отколкото тези по проформа. След преминаване през екрана за планиране се изпълнява следния алгоритъм:

- Извличане на необходимите данни от *vendors_customers* и *bovie_data*;
- Проверка дали документа вече не е бил създаден от *doc_recs*;

- Записва се самия документ в *doc_recs* и *doc_pozitions*. Проформата е със статус отворен. Този статус се променя от създаваните фактури към тази проформа;
- Запазват се позициите по дати на планиране в *pos_proforma_planirane;*
- Визуализира се самата проформа за печат;

Редактирането е реализирано на същия принцип, само че се прави промяна на позициите само на doc_recs (на основния запис на документа). Позициите на документа и тези за планирането, първо се изтриват и после се създават отново (*doc_pozitions, pos_proforma_planirane*). Също така се проверява дали няма необходимост от това да бъде променен статуса на проформата.

Референтни файлове:

- Създаване

Proforma_form1.php -> Proforma_form2.php -> Proforma_form2_2.php -> Proforma_form3.php

- Редактиране

Edit_Proforma_form0.php -> Edit_Proforma_form1.php -> Edit_Proforma_form2.php -> Edit_Proforma_form2_2.php -> Edit_Proforma_form3.php

Поръчка към доставчик

Поръчките към доставчик са документи, с които се възлага доставянето на даден компонент на доставчик (външен или вътрешен). Вътрешните производства на Боуви, също се явяват доставчици. По този начин може да се следи и какви материали има предадени при тях и какво трябва да бъде произведено. Трябва да се пишат поръчки за всички компоненти, който се очаква да бъдат доставени за да бъдат отчетени при планирането. При създаването си поръчката автоматично приема статус отворена и се затваря от складови разписки или митнически декларации за АУ или редовен внос. Няма нещо особено при създаването на документа. Отново се преминава през алгоритъм подобен на този при проформата:

- Извличане на необходимите данни от *vendors_customers* и *bovie_data*;
- Проверка дали документа вече не е бил създаден от *doc_recs*;
- Обновява се таблицата с номерата на компоненти на доставчиците или се въвежда нов запис ако там няма такъв. Таблицата е *component_vendor*;
- Записва се самия документ в *doc_recs* и *doc_pozitions.*
- Визуализира се самата поръчка за печат;

При редактиране, по-особеното е, че се гледа за евентуална промяна на статуса на поръчката, в зависимост от това какви са промените по нея.

Референтни файлове:

- Създаване

Bovie_PORACHKA_form1.php -> Bovie_PORACHKA_form2.php -> Bovie_PORACHKA_form3.php

- Редактиране

Edit_Bovie_PORACHKA_form0.php -> Edit_Bovie_PORACHKA_form1.php -> Edit_Bovie_PORACHKA_form2.php -> Edit_Bovie_PORACHKA_form3.php

Складова разписка

Складовата разписка е документ, с който в склада се завеждат компоненти и материали. Чрез нея се променя статуса на поръчките, от отворени на затворени.

В основният екран за създаване на складова разписка, има изборно поле за доставчик, данните, за което се зареждат от *vendors_customers*. При избор на доставчик в друго изборно поле се зареждат данните за отворените поръчки към този доставчик. Това става от таблици *doc_recs*. От това дали се попълва това поле за поръчка зависят две важни неща, а именно дали ще се затвори съответната поръчка или ще остане със статус отворена и дали ще имаме възможност да проследим произхода на елементите при необходимост. За проследяването е необходимо и полето за ЛОТ номер (уникален номер характеризиращ партидата на компонента).

Отново има същия стандартизиран начин за избиране или попълване на компонент за позицията. След като бъде натиснат бутона за продължение се преминава през следния алгоритъм:

- Извличане на необходимите данни от *vendors_customers* и *bovie_data*;
- Проверка дали документа вече не е бил създаден от *doc_recs*;
- Създаване на документ с *CreateDocument()* - таблици *doc_recs и doc_pozitions;*
- Въвеждане в склада на компонентите по позиции с *InputInStorage() - storage;*
- Промяна на статусите на поръчки с *ManageDocStatusPor() – doc_recs;*
- Визуализира се самата поръчка за печат;

При редактирането алгоритъма е същия като добавим това, че се връщат статусите на документите, затворени от тази складова разписка, преди другите действия. Внасят се новите количества в склада преди да се изнесат старите.

Референтни файлове:

- Създаване

Skladova razpiska_form1.php -> Skladova razpiska_form2.php -> Skladova razpiska_form3.php

- Редактиране

Edit_Skladova razpiska_form0.php -> Edit_Skladova razpiska_form2.php -> Edit_Skladova razpiska_form3.php

Протокол за качество

Протоколът за качество е документ, който има само една позиция. Неговата роля е да окачествени произведени от Боуви или от външен подизпълнител компоненти.

В основния му екран има:

- изборно поле за доставчик, в случая той се явява производителя на компонента, данните за което се извличат от *vendors_customers*;

- изборно поле за компонент, данните за което са извлечени от *components;*
- количество на компонента;
- поле за дата на документа;
- поле за поръчка към доставчика – данните за него се попълват в зависимост от избрания доставчик от *doc_recs*;

След натискане на бутона за продължение се изпълнява следния алгоритъм:

- Извличане на необходимите данни от *vendors_customers, components* и *bovie_data*;
- Проверка дали документа вече не е бил създаден от *doc_recs*;
- Създаване на документ с *CreateDocumentPPK()* - таблици *doc_recs и doc_pozitions;*
- Визуализира се информативна форма и се дава възможност при избор на бутона за печат да се покаже самия попълнен документ;

Референтни файлове:

- Създаване

BLANKA PROTOKOL ZA KACHESTVO_1.php ->
BLANKA PROTOKOL ZA KACHESTVO_2.php -> PPK.php

- Редактиране

Edit_BLANKA PROTOKOL ZA KACHESTVO_0.php ->
Edit_BLANKA PROTOKOL ZA KACHESTVO_1.php ->
Edit_BLANKA PROTOKOL ZA KACHESTVO_2.php ->PPK.php

Протокол за брак

Протокола за брак, също като този за качество, има само една позиция. Неговата роля е да се бракуват компоненти като се изнесат от склада.

В основния му екран има:

- изборно поле за доставчик, в случая той се явява производителя на компонента, данните за което се извличат от *vendors_customers*;
- изборно поле за компонент, данните за което са извлечени от *components;*
- количество на компонента;

След натискане на бутона за продължение се изпълнява следния алгоритъм:

- Извличане на необходимите данни от *vendors_customers, components;*
- Проверка дали документа вече не е бил създаден от *doc_recs*;
- Изваждане от склада на зададените количества;
- Създаване на документ с *CreateDocument()* - таблици *doc_recs и doc_pozitions;*
- Визуализира се протокола за брак за печат или съобщението за грешка при проблем;

При редактирането има една допълнителна стъпка за вмъкване в склада на това което стария протокол е бракувал.

Референтни файлове:

- Създаване

BLANKA PROTOKOL ZA BRAK rev 3_1.php ->
BLANKA PROTOKOL ZA BRAK rev 3_2.php

- Редактиране

Edit_BLANKA PROTOKOL ZA BRAK rev 3_0.php ->
Edit_BLANKA PROTOKOL ZA BRAK rev 3_1.php ->
Edit_BLANKA PROTOKOL ZA BRAK rev 3_2.php

Приемо-Предавателен протокол

Приемо-предавателният протокол служи за предаване на компоненти към производство или към фирма подизпълнител на дадена поръчка. От тези елементи се прави готовия компонент и той се заприхождава със складова разписка в склада.

Попълването на основния екран следва да започне от избора на доставчик. Данните за това изборно поле са заредени от vendors_customers и след като бъде избран посредством JavaScript в изборното поле за поръчката (заредено от *doc_recs*) остават само поръчките към този доставчик. Следва да се избере компонент чрез стандартния начин с бутона за избор или чрез написването на номера му в текстовото поле за целта. След което се избира номер на ЕАД или складова разписка. При изборното поле за ЛОТ номер се зареждат като възможности, само тези номера, които отговарят на избраното ЕАД или складова разписка и на съответния компонент, който определя позицията. След попълването на формата и натискането на бутона за продължение се преминава към следния алгоритъм:

- Извличане на необходимите данни от *vendors_customers* и *bovie_data*;
- Проверка дали документа вече не е бил създаден от *doc_recs*;
- Изваждане от склада на компонентите по позиции с O*utputFromStorage() - storage;*
- Създаване на документ с *CreateDocument()* - таблици *doc_recs и doc_pozitions;*
- Визуализира се протокола за печат;

Поради спецификата на предаване на елементи за производството на компонент е създаден и още един начин за работа с ППП, а именно чрез предаване по BOM (рецептурник). Това става като в допълнителни екрани се пита до какво ниво на разлагане да се достигне, обикновено първо (това е и стойността по подразбиране), по стандартния начин се избират компонентите и количествата им, а след натискане на бутона за продължение на следващия екран са заредени автоматично съставните на избраните компоненти, като количествата им се получават като умножение на тези от рецептурника и на това какво количество елементи се предава. Разбиването на съставни става с функцията GetComponentPart(). Количествата за еднаквите компоненти след разбиването се сумират. След това обработката продължава както при обикновеното създаване на ППП.

При редактиране процеса е същия, но преди другите действия се връщат в склада взетите компоненти.

Референтни файлове:

- Създаване

PPP_form1.php -> PPP_form2.php -> PPP_form4.php

- Създаване BOM

PPP_BOM_step0.php -> PPP_BOM_step1.php -> PPP_BOM_step2.php -> PPP_form4.php

- Редактиране

Edit_PPP_form1.php -> Edit_PPP_form1_next.php -> Edit_PPP_form2.php -> Edit_PPP_form3.php

Фактура

Това е така наречената US фактура. Тя е документ за плащане на изпълнена част от проформа-фактурата, който се изпраща на получателя на стоката (клиента). Фактурата винаги се отнася към дадена проформа, затова още на първия екран при създаването ѝ има задължително изборно поле с отворените проформи-фактури. Данните за това поле се взимат от doc_recs. След като е избрана проформа, на следващият екран автоматично от *doc_recs, doc_pozitions, measures u components* са попълнени всички полета на фактурата, включително цена и количество. Те могат да бъдат променяни. След натискане на бутона за продължение се изпълнява следния алгоритъм:

- Извличане на необходимите данни от *vendors_customers* и *bovie_data*;
- Проверка дали документа вече не е бил създаден от *doc_recs*;
- Създаване на документ с *CreateDocumentFact()* - таблици *doc_recs u doc_pozitions;*
- Промяна на статусите на проформа при необходимост;
- Визуализира се протокола за печат;

Референтни файлове:

- Създаване

BOVIEBGFACT_form1.php -> BOVIEBGFACT_form2.php -> BOVIEBGFACT_form3.php

- Редактиране

Edit_BOVIEBGFACT_form0.php -> Edit_BOVIEBGFACT_form1.php -> Edit_BOVIEBGFACT_form2.php -> Edit_BOVIEBGFACT_form3.php

BGN Фактура

BGN фактурата е българския еквивалент на разгледаната по-горе фактура. Тя няма отношение към склада или към промяна на статусите на други документи, а служи за обръщане на валутите към лев и за предаване на счетоводството. Използването и става предимно през функцията за копиране, в която се посочва US фактурата от която ще се вземат всички детайли и след обръщането на валутите към лева ще се визуализират. Данните се извличат от *doc_recs, doc_pozitions, components* и автоматично се попълват съответните полета в BGN фактурата. След натискане на бутона за продължение се прави

проверка дали такъв документ вече съществува и ако не документа се записва и се визуализира под форма за печат.

Референтни файлове:

- Създаване

BGFACT_form1.php -> BGFACT_form2.php -> BGFACT_form3.php

- Копиране

FACT_form1.php -> FACT_form2.php -> FACT_form3.php

- Редактиране

Edit_BGFACT_form0.php -> Edit_BGFACT_form1.php -> Edit_BGFACT_form2.php -> Edit_BGFACT_form3.php

4.2.3. Справки

Навсякъде, където се говори за полета въвеждане на дати се има в предвид формат на датата: дд.мм.гггг като за частта за година е допустимо да се въведат две цифри. Например 07 за 2007-ма година.

След това тези дати се променят във формат гггг-мм-дд посредством функцията convert_date_n_to_db('дата') за бъдат разбираеми за базата от данни.

Обратно навсякъде където се прочитат дати от базата данни и се визуализират после се използва функцията convert_date_db_to_n('db_дата').

Във всеки от екраните за избор (селекция), при натискане на бутона за продължение, се правят необходимите проверки, посредством JavaScript функции за коректността на въведените данни и за това, че всички задължителни полета за попълнени. Ако не е така се визуализира предупредителен прозорец и не се допуска преминаването напред.

При всяко обръщение към базата данни се прави проверка за върнатия резултат и при грешка се визуализира текст на екрана и изпълнението на програмата не продължава. Изпълнението, на която и да е функция без да има валидна сесия с потребителско име и парола е невъзможно заради проверката, която се прави в началото на всеки файл, чрез Check.php.

Складови Наличности

Движение на компонент

Справката дава възможност при избор на елемент, интервал от време и тип на документ, да се види участието на избрания компонент в документите.

В екрана за избор се извличат всички компоненти от таблицата *'components'*, с техните номера и имена. Дадена е възможност в поле за редактиране да се напише номера на желания компонент или чрез натискане на клавиша 'f', при активен елемент за избор на компонент, да се отвори помощна форма, с която да се улесни избора на компонент. В полето, където може да се напише номера на компонента може да се използва и знака '%',

за указване на множествена селекция. Например искаме да извадим движението на всички трансформатори от подгрупа 027. Тогава трябва да напишем 19-027%

В друг елемент за избор за заредени и всички типове документи от таблицата *'docs_type'*.

Останалите елементи от екрана са две полета за специфициране на интервал от време, за което да се направи избора на документи, в които този компонент участва. С натискане на бутона за продължение се зарежда резултата от справката.

Алгоритъмът през, който се преминава докато се стигне до визуализирането на резултата е следния:

- Структуриране на *'where'* клаузата за SQL заявка към базата данни, по зададените на екрана за избор критерии;
- Извличане на необходимите неща за визуализиране като например мерна единица и име на тип на документа (от предишната страница е предадено само по ID);
- Извличане на отрязъка от базата данни, по конструираната вече *'where'* клауза. Извлеченият списък е сортиран по тип на документа; Участват таблиците *'doc_pozitions, doc_recs, docs_type'*.
- Визуализиране на резултата, като в зависимост от типа на документите, се правят и суми по тип и обща сума, за цялата справка, по количества.

Ако са обхванати всички документи за този компонент крайната сума ще представлява наличността на компонента в склада.

Референтни файлове:

Show_movement_el.php -> Show_selected_movement_el.php

По днешна дата

Справката дава текущото състояние на склада по компоненти. В нея няма екран за избор.

Тя просто визуализира съдържанието на таблицата *storage* в табличен вид. Поради това, че таблицата storage се състои от полета индекси, SQL заявката включва и таблиците *'components'* и *'measures'*, за да може показаното на екрана да е в разбираем вид.

Референтни файлове:

Inventory_to_today.php

По фирми в склада

Това е изисквана от външните клиенти справка и затова е на английски.

При избор на доставчик, интервал от време и тип на документите дава наличните от този доставчик елементи, които са предани на „Боуви-България" ЕООД.

В екрана за селекцията има изборно поле за доставчик, отново полета за интервала от време, за който е желана справката и четири бутона за отметка (checkbox), чрез които

потребителят може да избере какви типове документи да се включат в селекцията и дали типа на документа да се вижда в генерираната справка.

Алгоритъма през, който се преминава докато се стигне до визуализирането на резултата е следния:

- Преминава през структуриране на *'where'* клаузата за SQL заявка към базата данни, по зададените на екрана за избор критерии;
- Извличат се името на доставчика от *'vendors_customers'* (от предишната страница е предадено само ID);
- От doc_recs, doc_pozitions, components, measures, currencies, docs_type, чрез конструираната *'where'* клауза се извличат необходимите данни. Сортирани са по номер на компонент. Започва визуализирането на таблицата като чрез двумерни масиви се правят суми по компонент за всички видове валути, за този компонент. Прави се и сума на така получените за компонентите суми за да се получи финално обща сума по всички видове валути, които се срещат в справката.

Референтни файлове:

Show_Comp_in_firms_form0.php -> Show_Comp_in_firms_form1.php

Митнически Декларации

Всички

Да показва всички митнически декларации за внос с техните позиции от елементи, и за всяка позиция списък с декларациите, с които са изнасяни количества от тези елементи;

Справката няма екран за селекция.

Първоначално се прави селекция от *doc_recs, doc_pozitions, components, measures* на всички документи от тип митнически декларации за АУ и за редовен внос и за техните позиции. Тази информация се обработва и съхранява в масив. За всяка от позициите на този масив се прави селекция, от *doc_recs, doc_pozitions_ref*, на митническите декларации за редовен износ, за износ за АУ или за митническа декларация за унищожение и резултата се съхранява в друг масив. Накрая се извежда резултата в табличен вид като за всяка декларация за внос има списък с позициите й (какви компоненти се внасят). За всяка от позициите, има списък с митнически за износ, с които тази позиция е изнасяна. Другите важни полета са внесените и изнесените количество и остатъчните количества по позиции на декларацията за внос.

Референтни файлове:

EAD_information.php

По документ

Както горната справка за всички декларации, но с възможност за избор на декларация;

Това, което я отличава от горната справка е екрана за избор, затова ще бъде разгледан само той. В него има изборно поле данните, за което са извлечени от *doc_recs*. Сортирани са по дата на документа и по номер. Изборното поле е задължително ако искаме да продължим.

След избора на декларация, се натиска бутона за продължение и се преминава към същият алгоритъм, но в първоначалната селекция има ограничение наложено от номера на избраната декларация.

Референтни файлове:

One_EAD_information_form1.php -> One_EAD_information.php

Документи

Селекция по всички възможни критерии, които притежава документ. Тя показва всички характеристики на документа и неговите позиции в списъчен вид;

Екрана за избор има следните възможности:

- Поле за номер на документ – в него се въвежда номер на желан документ като е възможно да се използва и символа '%' за множествен избор или ако не се знае точно номера на документа;
- Доставчик/Клиент – поле за избор, данните за което са извлечени от *vendors_customers;*
- Валута – поле за избор, данните за което са извлечени от *currencies;*
- Извършил операцията – поле за избор, данните за което са извлечени от *users;*
- Тип на документите – поле за избор, данните за което са извлечени от *docs_type;*
- Статус на документите – поле за избор, данните за което са извлечени от *customs_doc_status;*
- Полета за въвеждане на интервал от време;
- Полета за компонент – едно изборно с активирана функция *'f'* за помощ при търсенето и текстово поле за директно въвеждане на номер на компонент;
- Изборно поле за указване на начина на сортиране;
- Поле ограничаващо броя върнати резултати от справката (по подразбиране е 100).

С натискането на бутона за преминаване напред се изпълнява следния алгоритъм:

- Структуриране на *'where'* клаузата за SQL заявка към базата данни, по зададените на екрана за избор критерии;
- Извличане на данни за разбираемо показване на подадените с идентификатори параметри на селекцията;
- От *doc_recs, vendors_customers, customs_doc_status, users, currencies, docs_type*, чрез конструираната *'where'* клауза се извличат необходимите основни данни за документа. Сортирани са по избрания в екрана за селекция начин.

- Обхождане ред по ред горната селекция като за всеки ред (различен документ), от *doc_pozitions, components, measures* се извличат данните за позициите на документа. Ако в екрана за селекция е избран компонент тогава се филтрират позициите на документа само до тези, които съдържат този компонент или група компоненти.
- Показват се в списъчен вид всички данни.

Референтни файлове:

Show_documents.php -> Show_selected_documents.php

Елементи BOM

Справка, която показва за избран компонент, в BOM-овете (рецептурниците) на кои други компоненти участва.

В екрана за избор се извличат всички компоненти от таблицата *'components'*, с техните номера и имена. Дадена е възможност в поле за редактиране да се напише номера на желания компонент или чрез натискане на клавиша 'f', при активен елемент за избор на компонент, да се отвори помощна форма, с която да се улесни избора на компонент. В полето, където може да се напише номера на компонента може да се използва и знака '%', за указване на множествена селекция. Например искаме да извадим движението на всички трансформатори от подгрупа 027. Тогава трябва да напишем 19-027%.

С натискането на бутона за преминаване напред се изпълнява следния алгоритъм:

- Извличат се данни за разбираемо показване на подадените с идентификатори параметри на селекцията;
- От *bom_definition* се вади списък с позициите, в които този компонент фигурира.
- Обхожда се ред по ред горната селекция като за всеки ред, от *components, type_of_comp* се извличат данните за това кои е компонента в BOM-а, на когото е открит зададения в селекцията компонент.
- Намерените компоненти се визуализират в табличен вид.

Референтни файлове:

Elements_BOM.php -> Elements_BOM_show.php

Разходни норми

За избран компонент да се показват съставните му. Има възможност за разбивка до различни нива.

В екрана за избор се извличат всички компоненти от таблицата *'components'*, с техните номера и имена. Дадена е възможност в поле за редактиране да се напише номера на желания компонент или чрез натискане на клавиша 'f', при активен елемент за избор на компонент, да се отвори помощна форма, с която да се улесни избора на компонент. В полето, където може да се напише номера на компонента може да се използва и знака '%',

за указване на множествена селекция. Например искаме да извадим движението на всички трансформатори от подгрупа 027. Тогава трябва да напишем 19-027%

На този екран може да се укаже и до каква дълбочина да е показването. По подразбиране е едно. Това означава, че се показват само компонентите, от които се състои избрания компонент. Ако се избере две, тогава ще се покажат и съставните на тези от първо ниво и т.н..

При натискане на бутона за продължение се преминава към обработка на заявката. Извличат се данни за компонента, за визуализирането на заглавната част и се извиква функцията GetComponentPart('id_component','depth'), която връща масив с цялото дърво, от което се състои компонента и в зависимост от зададената в предишния екран дълбочина на показване, се визуализират тези компоненти които са на по-малка или равна на нея позиция. Различните дълбочини са отместени различно за по-добра читаемост.

Референтни файлове:

Razh_norma.php -> Razh_norma_show.php

Компоненти

Показва митническото движение за избран компонент. С кои декларации е внасян и с кои изнасян;

В екрана за избор се извличат всички компоненти от таблицата *'components'*, с техните номера и имена, и могат да бъдат избирани чрез изборно поле. Дадена е възможност чрез натискане на клавиша 'f', при активен елемент за избор на компонент, да се отвори помощна форма, с която да се улесни избора на компонент.

При натискане на бутона за продължение се преминава към следните стъпки:

- Вадят се от *doc_recs, doc_pozitions, currencies* необходимите данни като селекцията обхваща само митническите декларации за АУ и тези за редовен внос, като в позициите на документите трябва да фигурира посочения в екрана за селекция компонент. Резултата е масив съдържащ всички документи от изброените типове, които имат в позициите си този компонент;
- За всяка от горните позиции се прави селекция от *doc_recs, doc_pozitions_ref* за митнически декларации за редовен износ или за износ за АУ, или за митническа декларация за унищожение. Връзката с горните документи става чрез id_doc в *doc_pozitions_ref*; Резултата е масив съдържащ всички документи от изброените типове, които реферират към документите от горния масив;
- Събраните данни в масивите се визуализират.

Референтни файлове:

Component_information_form1.php -> Component_information.php

Разходна норма за Митница

За избрана митническа декларация за износ се показват материалите внесени за активно усъвършенстване, които тя сега експортира обратно. Показва и номера на фактурата за избраната митническа за износ.

На екрана за селекция има изборно поле, в което са заредени всички митнически декларации за износ (от таблица *doc_recs*) и бутон за продължение. След като бъде натиснат бутона се преминава към следния алгоритъм:

- За избрания документ се прави селекция от *doc_pozitions_ref, doc_recs*. В нея са филтрирани само референтните полета отнасящи се до митническите за АУ;
- За всяка от горните позиции от *doc_recs, doc_pozitions* се извличат данни и се създава масив, който съдържа изнесените неща от горната точка, с коя декларация са дошли (по компоненти);
- От *doc_pozitions* за избрания документ се взимат всичките му позиции;
- За всяка от позициите се обработват горе събраните данни и се визуализират. Междинно се извличат данни и за компонентите и за мерните единици от components и measures.

Референтни файлове:

Razhodna_norma_mitnica.php -> Razhodna_norma_mitnica_show.php

Проформа фактура

Дава възможност да се прегледат проформите по доставчик или по номер на документ за даден интервал от време. Резултатът се визуализира в таблична форма, като под всяка позиция на проформа се изрежда списък с фактурите, които са изнасяли от тази позиция. Ясно се вижда и остатъка по позиции на проформата и като количество и като цена.

В екрана за селекция от vendors_customers са заредени данните в изборно поле за специфициране на доставчик, за когото да е справката. Има текстово поле за въвеждане на номер на документ, където е възможно използването на знака ‘%’, за множествена селекция на документи и 2 полета за определяне на интервал от време.

След натискане на бутона за продължение се пристъпва към изпълнението на следния алгоритъм:

- Формира се „*where*” клаузата за *SQL* заявката;
- От таблица *vendors_customers* се извличат данни за доставчика ако е избран такъв;
- От doc_recs, components, doc_pozitions, measures, currencies се извличат данните чрез създадената „*where*” клауза;
    - За всеки ред от селекцията се преминава през намиране на остатъчното количество по компонент и проформа. Това става чрез заявки към базата данни (таблици *doc_pozitions, doc_recs*) от типа *select sum(quantity)*;
    - За всяка от позициите се извличат данни за фактурите реферирващи към тази позиция от таблици *doc_pozitions, doc_recs;*
    - Визуализират се данните.

Референтни файлове:

Show_Proforma_form0.php -> Show_Proforma_form1.php

Поръчка към доставчик

Същата като за проформа фактурата, но за поръчка за доставка и съответно складови разписки или митнически декларации, които променят остатъчните количества и цени на позициите, на поръчката за доставка.

В екрана за селекция от *vendors_customers* са заредени данните в изборно поле за специфициране на доставчик, за когото да е справката. Има текстово поле за въвеждане на номер на документ, където е възможно използването на знака '%', за множествена селекция на документи и 2 полета за определяне на интервал от време.

След натискане на бутона за продължение се пристъпва към изпълнението на следния алгоритъм:

- Формира се „*where*" клаузата за *SQL* заявката;
- От таблица *vendors_customers* се извличат данни за доставчика ако е избран такъв;
- От doc_recs, components, doc_pozitions, measures, currencies се извличат данните чрез създадената „*where*" клауза и при ограничението, че става дума за поръчки;
    - За всеки ред от селекцията се преминава през намиране на остатъчното количество по компонент и поръчка. Това става чрез заявки към базата данни (таблици *doc_pozitions, doc_recs*) от типа *select sum(quantity)*, за документи от тип митническа декларация за АУ, митническа за редовен внос и складова разписка;
    - За всяка от позициите се извличат данни за входните параметри реферирящи към тази позиция, от таблици *doc_pozitions, doc_recs;*
    - Визуализират се данните.

Референтни файлове:

Show_Bovie_PORACHKA_form0.php -> Show_Bovie_PORACHKA_form1.php

Показване компонент

По избран компонент показва характеристиките му, както и неговите съставни компоненти.

На екрана за избор има следните полета:

- Изборно поле за група, данните за което са извлечени от *comp_groups;*
- Текстово поле за въвеждане на подгрупа;
- Текстово поле за въвеждане на номер на компонент, в което е позволено използването на символа *'%'*, за множествена селекция;
- Текстово поле за описание на компонент;

- Изборно поле за тип на компонента, данните за което са извлечени от *type_of_comp;*
- Изборно поле за мерна единица, данните за което са извлечени от *measures;*
- Текстови полета за ревизия и дата на ревизия;
- Поле ограничаващо максималния брой изведени резултата. По подразбиране със стойност 100.

При натискане на бутона за продължение се преминава към следния алгоритъм:

- Формира се „*where*" клаузата за *SQL* заявката;
- Чрез тази клауза, от таблици *components, measures, type_of_comp, comp_subgroups, comp_groups*, се извличат основните данни за компонента;
- Данните се визуализират;
- По идентификатора за BOM от *bom_definition* се извличат данните за рецептурника на компонента ако има такъв;
    - За всеки ред от него се извличат данните за компонента (от таблици *components, measures, type_of_comp*);
    - Данните се визуализират.

Референтни файлове:

Show_comp.php -> Show_selected_comps.php

Планиране

Планирането е справка, която позволява автоматизиране на процеса за поръчване на елементи. Анализирайки необходимостта за компоненти, които трябва да бъдат доставени на клиентите, отворените поръчки, които са договори за доставка на компоненти за Боуви и наличностите в склада, се прави списък с компоненти, които не достигат. По конкретно, за това какво трябва да се достави, се гледа таблицата *pos_proforma_planirane*. В нея освен идентификатори на желаните компоненти има и срок за доставка и количество.

При избиране на планиране се визуализира екран за въведение, където трябва да се зададе крайна дата, до която се планира и производствен период в дни. Планирането обхваща всички документи до зададената дата минус броя дни за производство (lead time).След въвеждане на двете задължителни полета се натиска бутона за продължение. Следва следния алгоритъм:

- Зарежда се масив със всички налични компоненти в склада, от таблица *storage;*
- Зарежда се масив с отворените поръчки към доставчик с дата на получаване по-малка от зададената дата минус броя дни за производство (lead time). Еднаквите позиции се сумират. Особеното е че ако има поръчка за доставяне на сложно изделие не се смята, че неговите съставни вече са налични;
- Прави се проверка дали поръчките не са частично изпълнени и ако са към масива се добавят само нещата, които все още не са доставени. Проверката става като се анализират входящите документи към склада за тези компоненти;

- Следва проверка на отворените проформи от *doc_recs, pos_proforma_planirane.* За това дали позицията вече не е частично изпълнена се анализират фактурите за дадената проформа (*doc_recs, doc_pozitions*);
- Пресмятане на недостига чрез обхождане на масивите;
- Визуализиране на всички компоненти за изпълнението на отворените позиции по проформи. Визуализират се дори тези компоненти, които са налични. За тях в графата необходимо количество стои нула. За всяка проформа има бутон за планиране, натискането на който отваря нов прозорец.

След като е натиснат бутона за планиране, избраните с отметка позиции, които искаме да поръчаме се генерират в табличен вид, групирани по евентуалните поръчки. Групирането става като се чете от таблици *vendors_customers* и *component_vendor* кой доставчик последно е доставял от необходимия елемент. За всеки доставчик има отделно бутон генерирай, за създаване на поръчката. След натискането му се преминава към стандартно създаване на поръчка (обяснено в т. Документи), но самата форма е вече попълнена с желаните данни. Остава само да се напише цената за доставка.

Референтни файлове:

Planning_step1.php -> Planning_step2.php -> Planning.php -> CreatePorychka.php -> Bovie_PORACHKA_form3.php

Проследимост

По избран протокол за качество се генерира списък с произхода на съставните на окачествяваното изделие. При наличие на сложен компонент може да се извика справката рекурсивно за него. Това позволява да се проследи даден компонент от кои партиди елементи е произведен и при евентуален проблем да се открие източника му. За да е възможно това трябва да бъде спазван стриктно следния бизнес процес:

- Когато има необходимост от произвеждането на даден детайл се пише поръчка за доставка. Независимо дали става дума за вътрешното производство или за външна фирма подизпълнител;
- При предаване на необходимите елементи за производството на поръчания компонент в приемо-предавателния протокол се пише както номера на поръчката, така и номера на входящия документ, с който този елемент е влязъл в склада. Ако има наличен ЛОТ номер се пише и той;
- Пише се протокол за качество за готовото изделие, в което също се пише номера на поръчката, по която е произведен компонента;
- При завеждането в склада на готовия елемент отново се пише номера на поръчката, а за ЛОТ номер се пише номера на протокола за качество.

  Така цикъла е затворен и има пълна проследимост на компонентите.

В екрана за селекция има изборно поле, в което от *doc_recs* са заредени протоколите за качество. В него е валидна и функцията ‘f’ за улесняване на избора. Другото поле в екрана е текстово и позволява написването на номера на протокола за качество директно в него. В него не е позволено въвеждането на ‘%’ за множествена селекция.

След като бъде избран валиден протокол за качество (JavaScript функция се грижи това да е станало) и при натискане на бутона за продължение се минава към изпълнението на следния алгоритъм:

- От doc*_pozitions, components, doc_recs, vendors_customers, measures* се извличат всички данни за протокола за качество избран в екрана за селекция;
- Визуализират се основните данни за документа;
- В масив се зарежда първо ниво на съставните на компонента, за който е този протокол за качество. Това става като се четат таблици *components, type_of_comp, measures*;
- По поръчката се намират Приемо-предавателните протоколи, частите за които са изписани по тази поръчка. Това става като се четат таблици *doc_pozitions, components, measures, doc_recs, docs_type*;
- Прави се проверка за това дали броя от горната селекция съответства на тези от BOM. Ако не, се изкарва предупреждение;
- Дори при несъвпадение се визуализират позициите извлечени чрез селекцията по поръчката;
- Ако има въведен номер на ЕАД се визуализира;
- Ако има ЛОТ номер се визуализира. Ако той е номер на протокол за качество на Боуви, се прави като хипервръзка, за да може с едно кликване с мишката да се извика същия екран. но за новия протокол за качество.

Референтни файлове:

Show_LOT_form1.php -> Show_LOT_form2.php

4.2.4. Потребители

До функциите за работа с потребители има достъп само потребителя bovie. След като даден потребител остави „отпечатък" в системата, той не може да бъде изтрит, а само блокиран. Например ако създаде документ в таблицата *doc_recs* вече е записано, че потребител с това ID е създал документа и при опит да бъде изтрит, заради ограничаващите условия на базата данни, това няма да бъде позволено. Таблицата, в която се пазят данните за потребителите е *users*. В нея са предвидени полета за разграничаване на правата на потребителите до такива, които имат права за четене и промени и такива, които имат права само за четене. Паролата за достъп е кодирана с алгоритъма MD5. По този начин се избягва дори при компрометирана база да се разберат паролите и съответно да има неоторизиран достъп. Ако даден потребител е блокиран, това означава че той няма да може да влезе в системата.

В точката за базата данни е описано какви полета има в таблицата потребители. Функциите, които могат да се правят с потребителите са: дефиниране, редактиране, блокиране и изтриване;

Референтни файлове:

§ Дефиниране

Create_user.php -> User_save.php

§ Редактиране
Edit_user.php -> User_edit1step.php -> Edit_user_save.php

§ Блокиране
Lock_user.php -> Lock_user_save.php

§ Изтриване
Delete_user.php -> Delete_user_save.php

4.2.5. Доставчици/Клиенти

В системата не е предвидено разграничаване на фирмите в зависимост от това дали са доставчици или клиенти. Таблицата, където се съхраняват основните им записи, е *vendors_customers*. В точката за базата данни е описано с подробност каква е структурата и.

Функциите, които могат да се изпълняват при оперирана с основния запис на доставчик са: създаване, редактиране, изтриване и показване. Няма нищо особено по-екраните освен попълването на полетата и няма да бъдат разгледани подробно.

Веднъж използван при създаването на документ, доставчик не може бъде изтрит, за да се избегне неконсистентност в базата данни.

Друго особено нещо е, че вътрешните производства на Боуви също трябва да бъдат дефинирани като доставчици. Така целия процес се унифицира.

Референтни файлове:

§ Създаване
Create_vendor.php -> Vendor_save.php

§ Редактиране
Edit_vendor.php -> Vendor_edit.php -> Vendor_edit1.php

§ Изтриване
Delete_vendor.php -> Vendor_delete.php

§ Показване
Show_all_vendors.php

5. Тестване и внедряване

Целта на всяка създавана система е да бъде полезна и качествена. Доброто тестване е от изключителна важност.

Според [2]:

„Качествен е този програмен продукт, който удовлетворява формулираните към него изисквания.

Всяко отклонение от изискванията се нарича дефект.

Дефектите обикновено се дължат на една или няколко грешки.

Под грешка се разбира неправилност, отклонение или неволно преиначаване на обект или процес.

В зависимост от това, в какъв софтуерен продукт се откриват, грешките могат да бъдат грешки в проекта, в програмата, в документацията, в тестовите данни и т. н.

Според Майерс в програмата има грешки, ако тя не изпълнява това, което потребителят разумно очаква от нея.

Грешките могат да бъдат:

- *първични* - неправилности в текста на програмите, подлежащи на непосредствено коригиране;
- *вторични* - изкривявания на получените резултати (например зацикляне, дължащо се на първична грешка - непроменяне на стойността на управляващата променлива за цикъла в неговото тяло).

Има два основни подхода за тестване: структурен и функционален.

Целта на структурното тестване е да се изберат такива тестови данни, че да се осигури преминаване през всички програмни части на системата.

При функционалното тестване се проверява правилността на реализираните основни функции.”

В складовата и документираща система са използвани и двата подхода за тестване, в зависимост от конкретните условия. При по сложни конструкции където преминаването през разклоненията се определя по сложен начин и зависи от много фактори, е правено тестване с такива данни, че да се осигури проверка за това дали се преминава през всеки клон.

В зависимост от избраните тестови данни и очаквани резултати тестването се дели на:

- *детерминирано тестване* - при зададени входни данни напълно е определено какви трябва да бъдат получените резултати;
- *стохастично тестване* - тестваните данни са случайни величини с определено разпределение и се знае разпределението на получаваните резултати.

В разработваната система е използвано само детерминирано тестване;

В зависимост от целта тестването може да бъде:

- *за доказване на експлоатационната годност* - проверява се дали програмният продукт е работоспособен;
- *за атестация* - т. нар. пускови изпитания, след успешното завършване на които програмният продукт може да се разпространява;
- *за пълна функционална проверка* - дали реализираните функции съответстват на формулираните изисквания;

- за проверка на *специални програмни свойства,* например надеждност или преносимост;
- *за проверка на нови свойства или функции* - реализира се след внасяне на изменения в създадения програмен продукт и се нарича регресионно тестване;
- *за проверка на работоспособността на системата в реални потребителски условия* - изследва се времето за отговор на системата, продължителността на входно-изходните операции, изчислителните ресурси; качеството на потребителския интерфейс и други.

В зависимост от това, кой извършва тестването, то може да бъде:

- *вътрешно тестване* - от самите разработчици. Всеки програмист проверява правилността на създадените от него програми. Преимущество на този подход е, че не е необходимо разучаване на програмите. Недостатък е, че програмистите имат склонност към надценяване на възможностите си и увереността, че не допускат грешки, а това пречи на системното тестване. Форма на вътрешно тестване е т. нар. *peer review* - проверка, при която програмистите са разделени на двойки и всеки проверява програмите на партньора си.
- *независимо тестване* - от експерти, които не са участвали в разработването на програмите. Обикновено това са членове на групата по тестване, действащи в софтуерната фирма.
- *сертифициране* - проверка и издаване на сертификат, удостоверяващ наличието на определени свойства на ПП. Сертифицирането може да се извършва само от фирми, притежаващи съответния лиценз.

По този критерии е правено само вътрешно тестване. Както и тестване от страна на потребителите.

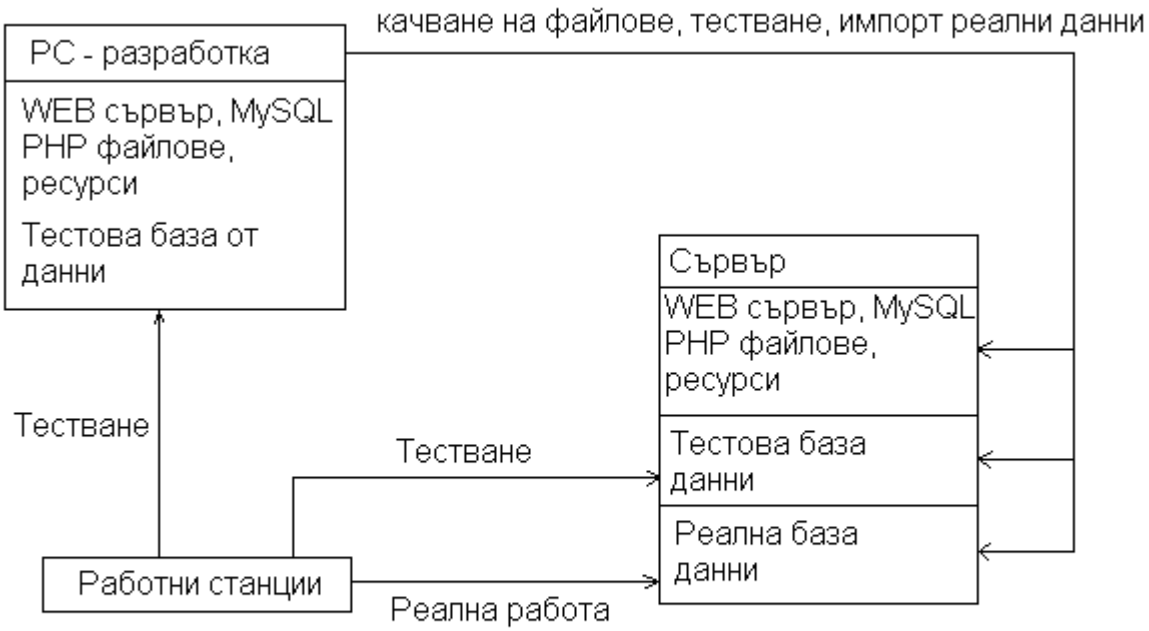
Инструменталният продукт, използван за подпомагане на процеса за настройване и откриване на грешки при разработката на системата е ZendStudio 5.0. Чрез него значително се улеснява откриването на грешки в по-сложни програмни конструкции, особено на фона на това, че алтернативата е отпечатване на екрана на стойностите на наблюдаваните променливи.

5.1. Тестване коректността на реализацията

Системата е изпробвана за работа с три различни web-клиента, а именно:

- Internet Explorer версии 6, 7;
- Mozila Firefox версия 2;
- Avant Browser версия 10.

По време на разработването на системата е използвана следната схема за гарантиране на качеството – фиг. 5.1.1



Фиг. 5.1.1. Схема за гарантиране на качеството

При разработката се изпробва цялата функционалност в момента на написването й. След което се оставят и потребителите да се убедят, че тя отговаря на заданието и на техните нужди. Още на този етап те правят забележки, които се отстраняват. След това файловете се качват на работния сървър, където е създаден механизъм за работа и с реалната база и с тестова такава, която често бива препокривана от реалната за осигуряване на по-добра среда за тестване. Това дали ще влезеш в тестовата система или в реалната, зависи от това с какво потребителско име ще се представиш. Така на потребителите е предоставена двойна възможност да се убедят в качеството на това, което им се предлага и да участват интензивно и да носят своята отговорност при създаването на продукта.

Освен това след първоначалното захранване с данни на системата са създадени няколко начина за проверка на верността на алгоритмите. Такъв например е теста, които изследва всички документи и по тях, за всеки компонент намира какво би трябвало да е текущото му състояние в склада. Този резултат се сравнява с таблицата storage и ако има разминаване се визуализира.

5.2. Тестове за производителност

Системата се разработва върху Windows базирана работна станция като тестовата система е с Apache WEB Server Version 1.3.34, PHP Script language Version 4.4.1, MySQL Database Version 5.0.16. Инсталирано е върху PC - AMD Athlon XP 2000+ на 1.6GHz с 512 MB RAM.

Първоначално при разработката бяха използвани изборни полета за всяка позиция на документите, но поради голямото количество компоненти, които се зареждат за избор (около 3000) и при голям брой позиции на документ (над 50), web клиента изразходва повече от 100MB RAM и съответно започва да се държи латентно. Това наложи преработването на този начин за избор на компонент и сега това става в допълнителна форма при натискането на бутон.

При тестването не са използвани никакви инструменти за изследване на производителността, а само стрес тестове на системата за големи отрязъци от данни и справки.

При работата на локалната работна станция с характеристиките описани по-горе най-бавното време за показване на справка е по-малко от 10 секунди, при обработка на над 2700 документа, които имат повече от 6500 позиции.

Техническо описание на сървъра където работи реално системата е:

Процесор: 2.4 GHz. Celeron
RAM: 256 MB
MB: ASUS P4
HDD: 2x200 GB, направени в режим RAID 1 (огледални)

При Интернет връзка с горепосочения сървър забавянето идва от необходимостта от предаване на голямо количество информация по мрежата и на практика зависи само от пропускливостта на мрежата. Най-тежката справка преминава за по-малко от 4 минути. Забелязва се и обичайната латентност при работа с Интернет страници, но тя не е нещо неочаквано поради гореспоменатата причина.

Още в изискванията за системата е заложено че ще се работи основно в локална мрежа, а възможността за работа и през Интернет е само допълнителна.

При тестването на различните функционалности, на различни работни станции, беше установено, че при обработка на големи документи с повече от 50 позиции, при компютри с по-малко от 256 MB RAM и относително бавни процесори, се получават по-големи забавяния. Това се обяснява с неспособността на web клиента, да се справи достатъчно бързо със зареждането на голямото количество информация, подавано му от сървъра.

5.3. Внедряване

Първоначално беше създадена базата данни през средата на Eclipse с помощта на добавката за нея clay. Тя предоставя възможност за визуално и интерактивно създаване не само на таблиците с техните полета и характеристики, но и на връзките и ограничителните условия между таблиците. Позволява също така от създадения модел да се генерират SQL заявки, които да го пресъздадат в реална база. Тя беше създадена с помощта на phpmyadmin database administrator.

С данни бяха заредени статичните таблици като тези за мерните единици, валути, типове на документи и т.н.

След първоначалното създаване на интерфейса и част от функционалността за работа с основните данни на доставчици, потребители, компоненти и т.н. се въведоха основните данни за групи, подгрупи, компоненти посредством SQL заявки. Те бяха автоматично генерирани от подадените от Боуви таблици на excel, посредством макроси. Така създадените заявки се изпълниха отново през phpmyadmin, като беше следвана последователност на изпълнение, такава че да не се нарушат ограничителните условия.

Например ако се опитаме да заредим данни в таблицата components преди да са били заредени такива в таблиците за групи и подгрупи (comp_groups и comp_sungroups) ще се върне съобщение за грешка, че няма такива индекси.

След това последователно са добавяни функционалносттите след тестване на всеки модул, до финалното завършване и на последните функции, със забележката че системата ще продължи да се развива със следващи версии, докато предишните са в експлоатация.

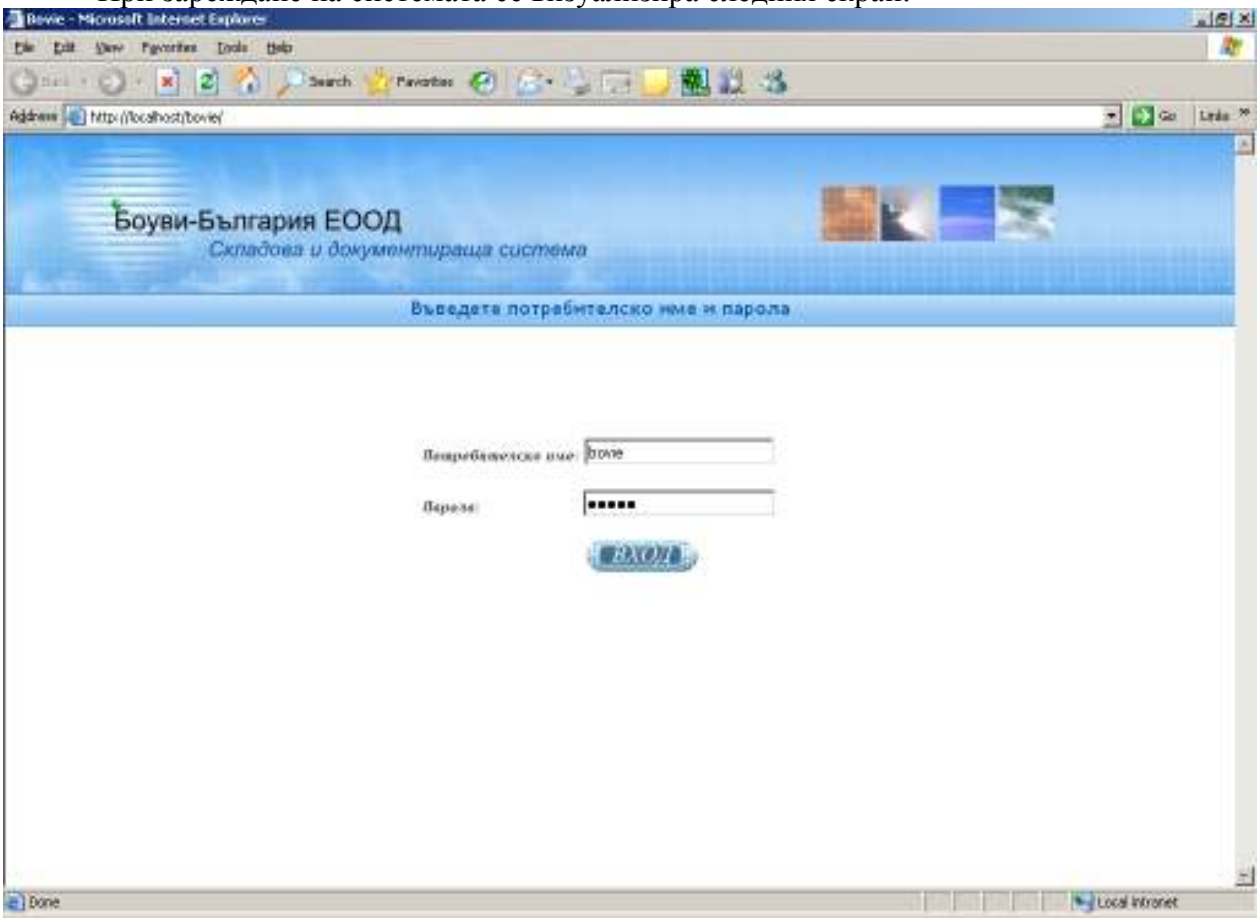
Макар и да няма създадена инсталационна процедура, не е никак трудно системата да бъде преместена на друг хардуер или да се направи допълнително внедряване. За тази цел се прави извличане на базата данни под формата на SQL заявки, копира се папката където са php файловете и другите ресурси, на новото място, създава се база със същото име за да не се налага промяна на файла който осигурява връзката с базата данни и се изпълняват SQL заявките. Така получената система е абсолютно копие на предишната.

6. Ръководство за потребителя

6.1. Общи положения

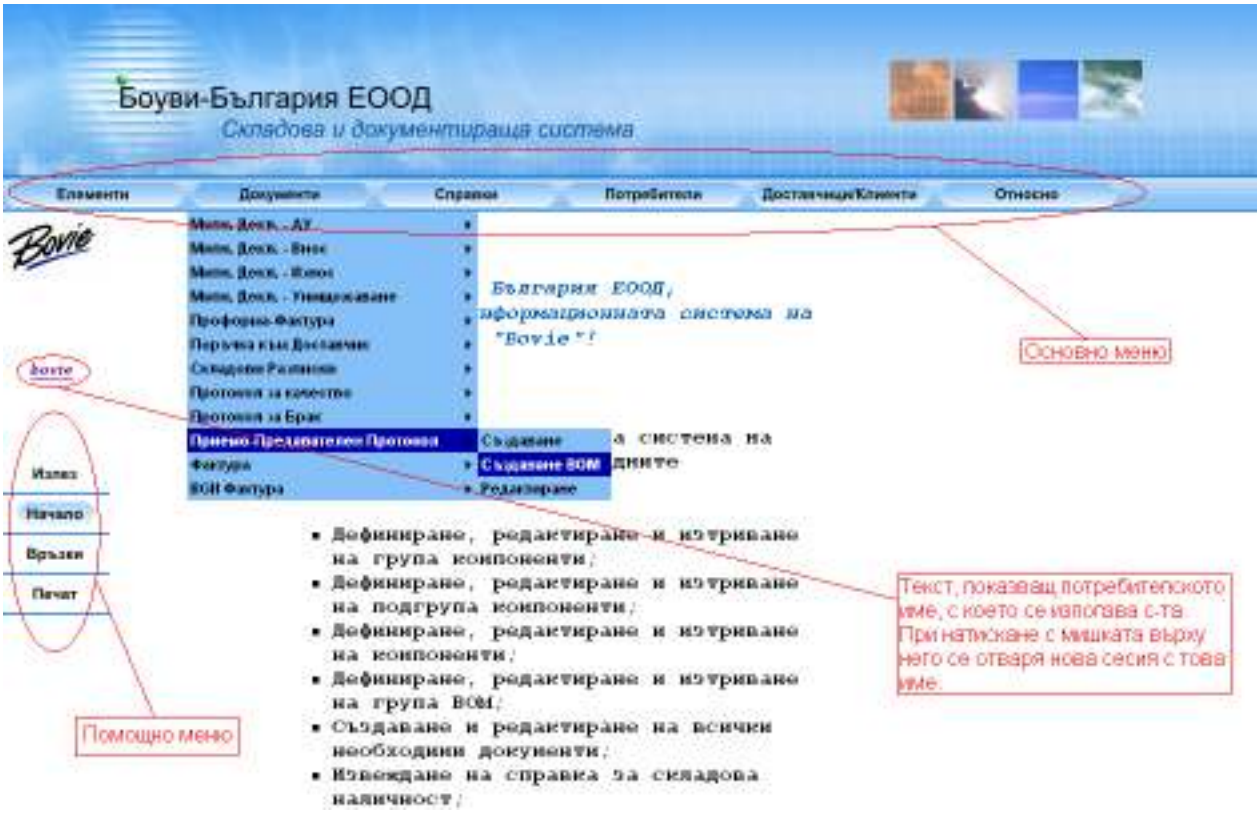
В тази точка ще бъдат разгледани най-основните екрани, повечето от които се повтарят във функционалностите. Целта е не да се направи подробно описание на всеки екран и бутон, а да се придобие представа за възможностите на интерфейса и как да бъдат използвани те.

При зареждане на системата се визуализира следния екран:



В него се попълват потребителското име и паролата и се натиска бутона за вход или клавиш ENTER.

Показва се основния екран на системата.



С червено са отбелязани основните компоненти на този екран. Те са общи за цялата функционалност. В зависимост от къде се намираме се променя основната част от екрана. Менютата дават по-голямата част от навигацията. При обхождането им може да се кликва с мишката върху желаното разклонение и така не прескача при преминаване неволно в областта на някое от съседните разклонения.

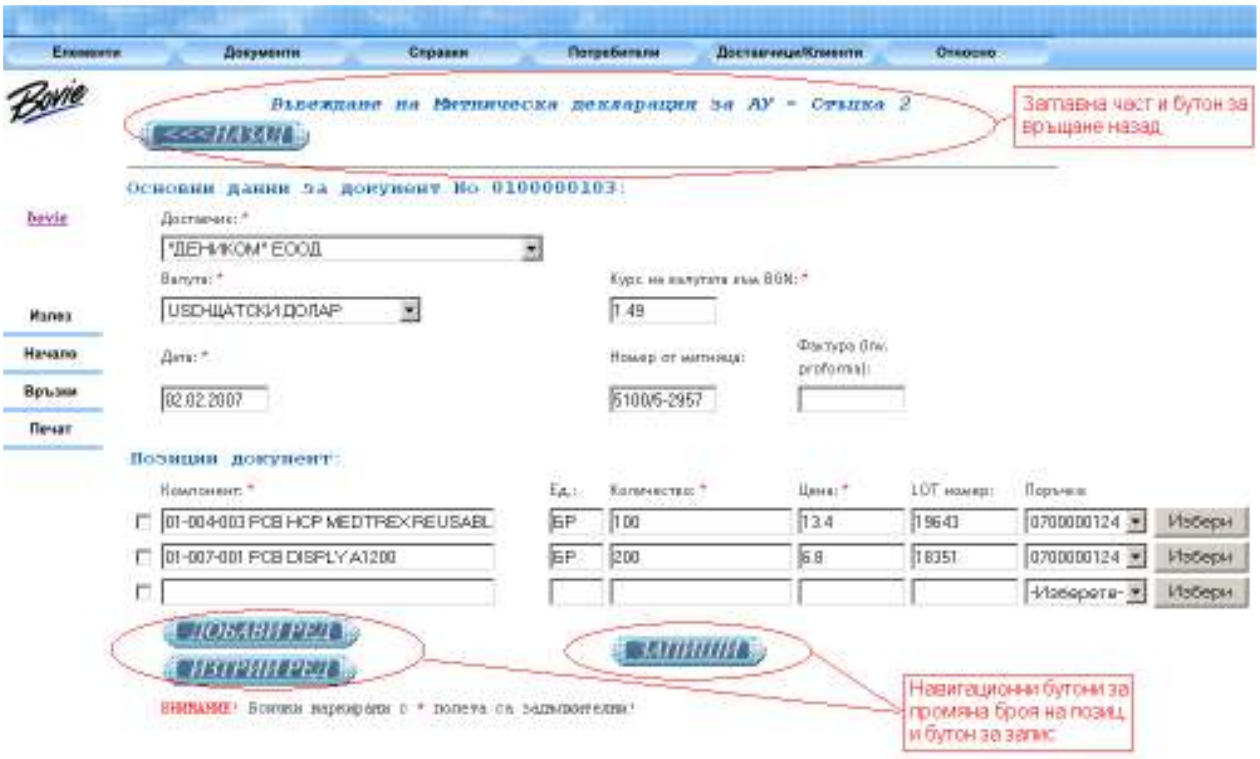
Предвиден е и още един начин за достъп до цялата функционалност, а именно чрез избиране на препратката Връзки във вертикалното меню. Там се визуализира списък с връзки към всичко, което предлага хоризонталното меню.

6.2. Документи

При създаването на всички документи, с изключение на Протокол на качество и Протокол за брак, в първоначалния екран има поле за брой позиции, които ще има документа. Например за митническа декларация за АУ екрана е:

Това спестява натискането на бутона за добавяне на позиция към документа на следващия екран. В горния пример може да се види и символа ‘*’. Както и пояснителен запис за ‘внимание’. Двете се срещат в почти всяка страница. Със ‘*’ се указват задължителните полета за попълване. Ако са оставени без стойност системата не допуска преминаването към следващия екран.

Има два вида дизайн на основните екрани за въвеждане на документи. Единият тип е, когато няма изискване за печат на бланка. При него полетата са подредени в табличен вид, като отгоре на страницата са тези за основния запис на документа (например доставчик, валута и т.н.), а по-надолу са позициите на документа. В долния екран е показан пример за попълване на митническа декларация за активно усъвършенстване.

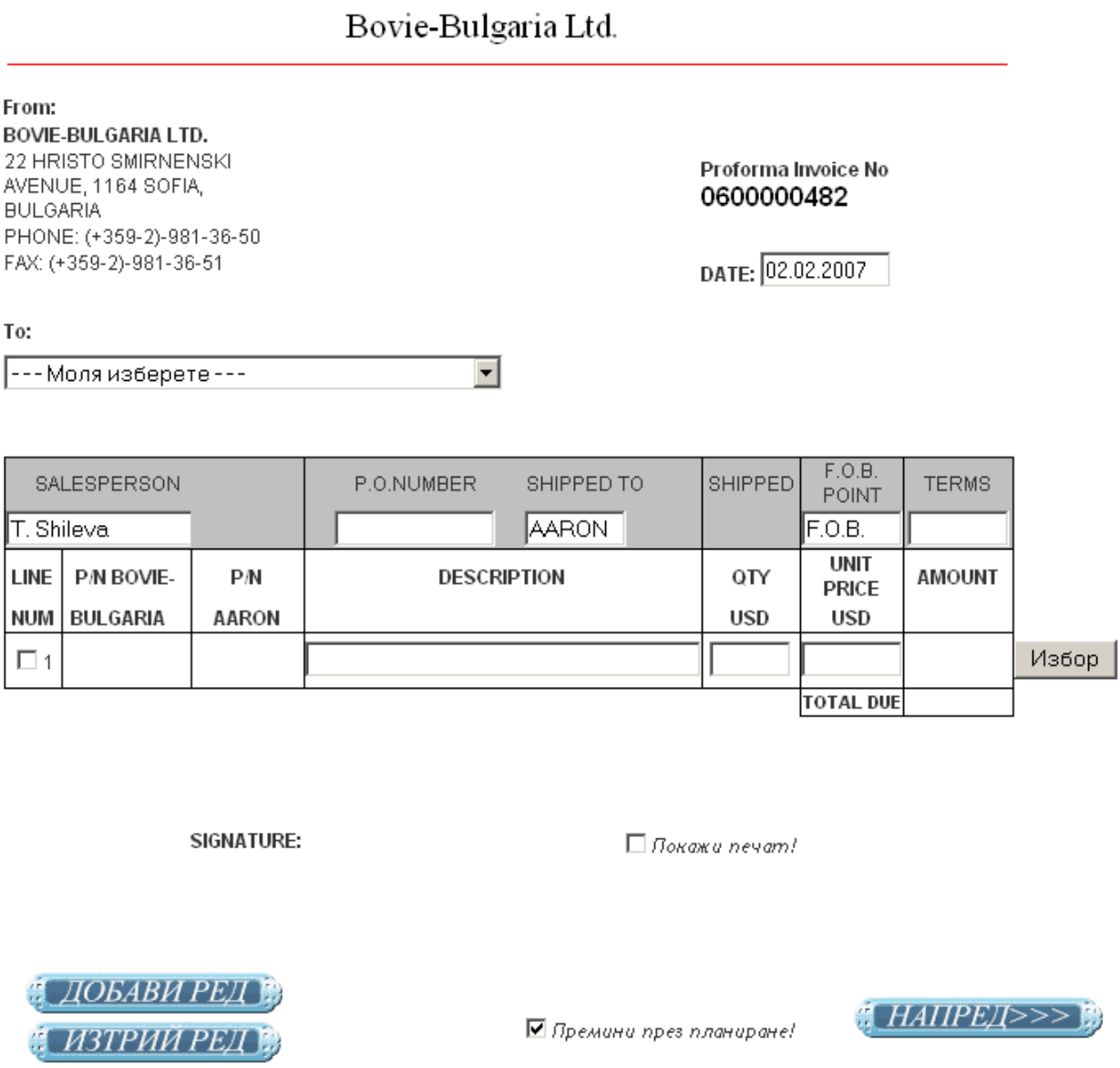


Заглавната част и бутона за връщане са характерни за всяка страница от системата, а бутоните за добавяне и изтриване за функционалности, в които има позиции на документите. При натискане на бутон за добавяне на ред се визуализира празен такъв най-отдолу, а ако искаме да изтрием дадена позиция трябва да се избере отметката най-отляво на реда с позицията, която искаме да изтрием и да се натисне бутона за изтриване.

Важно е да се отбележи, че при всички документи, в които фигурират полета за избор на доставчик и на поръчка, първо трябва да се избере доставчика, което зарежда отворените поръчки за този доставчик в полето за избор на поръчка. При не избран доставчик, в полето за избор на поръчка няма стойности.

Във всеки документ има поле за дата, което при влизане в страницата има за стойност днешна дата, но може да бъде променяно. То трябва да е във формат дд.мм.гггг. Допустимите разделители са ‘/.-’.

При вторият вид дизайн за създаване на документ водещото е самата бланка. Полетата за попълване не са извадени отделно, а са на местата където седят в бланката. Това помага за лесното свикване да се работи с документа (същите бланки са използвани и преди), както и за по-добра ориентация е него. Пример за това е долния екран за създаване на проформа фактура.



Отметките в горния екран играят ролята на избор на разклонение на програмата или за указване на начин на показване. В случая отметката премини през планиране, дава възможност за избор на потребителя дали да въведе данни за доставка сега или на по-късен етап, а отметката за печат, дава възможност за избор дали на резултатната бланка (показана по-долу), да се визуализира печат или не.

След попълване на формата и продължаване напред, документа се записва и се визуализира попълнената бланка, готова за печат.

Bovie-Bulgaria Ltd.

From:
BOVIE-BULGARIA LTD.
22 HRISTO SMIRNENSKI AVENUE, 1164 SOFIA, BULGARIA
PHONE: (+359-2)-981-36-50
FAX: (+359-2)-981-36-51

Proforma Invoice No
0600000482

DATE: 02.02.2007

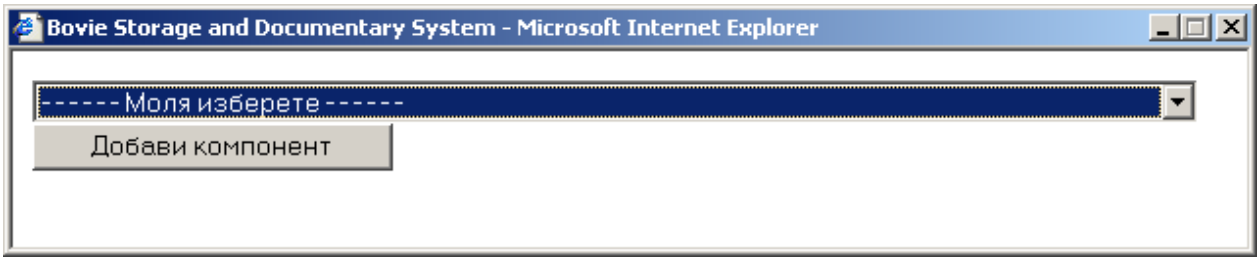
To:
"ДЕНИКОМ" ЕООД
кв."Слатина"
PHONE: 9714700

SALESPERSON T. Shileva			P.O.NUMBER SHIPPED TO AARON	SHIPPED	F.O.B. POINT F.O.B.	TERMS 12.12.2006
LINE NUM	**P/N BOVIE-BULGARIA**	**P/N AARON**	**DESCRIPTION**	**QTY USD**	**UNIT PRICE USD**	**AMOUNT**
1	22-003-001	22-003-001	EPROM, PROGRAMMED A1200	3	100.00	300.00
					TOTAL DUE	**300.00**

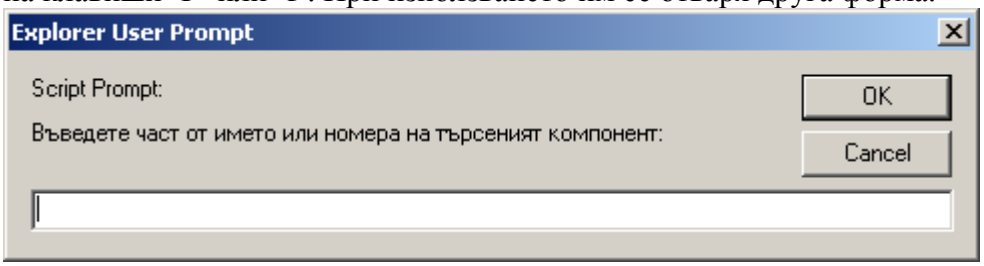
SIGNATURE:

Начините на попълване на компонентите за позиция във всеки от документите са следните:

- Директно се попълва номера на компонента в текстовото поле;
- Настиска се бутона за избор срещу желаната позиция. Той визуализира следната форма:

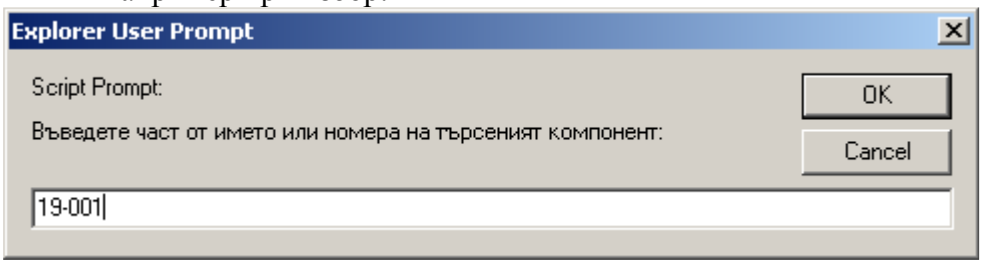


В нея, в полето за избор има списък с всички дефинирани компоненти, подредени по номер. Дадена е и възможност за допълнително улесняване на търсенето чрез натискане на клавиши ‘F’ или ‘f’. При използването им се отваря друга форма.

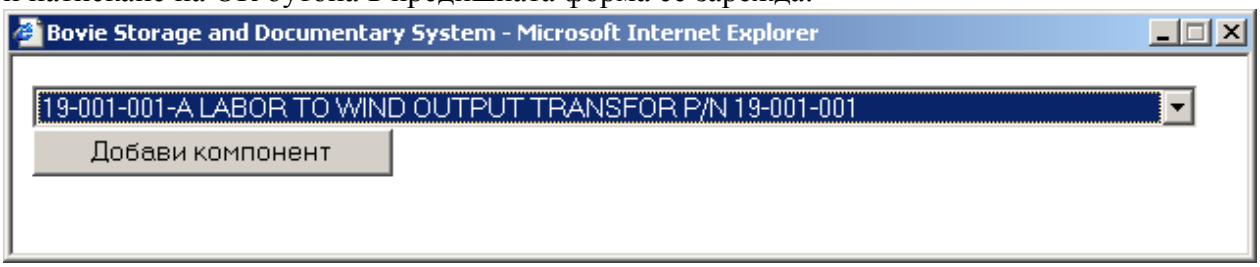


В текстовото поле се пише номера на компонента, до където е известен или например само номер на група (първите две цифри от номера на компонента). След като бъде написан част от номера и се натисне OK бутона в предишната форма избран елемент става първия такъв, който отговаря на зададения номер.

Например при избор:



и натискане на OK бутона в предишната форма се зарежда:



При натискането на бутона 

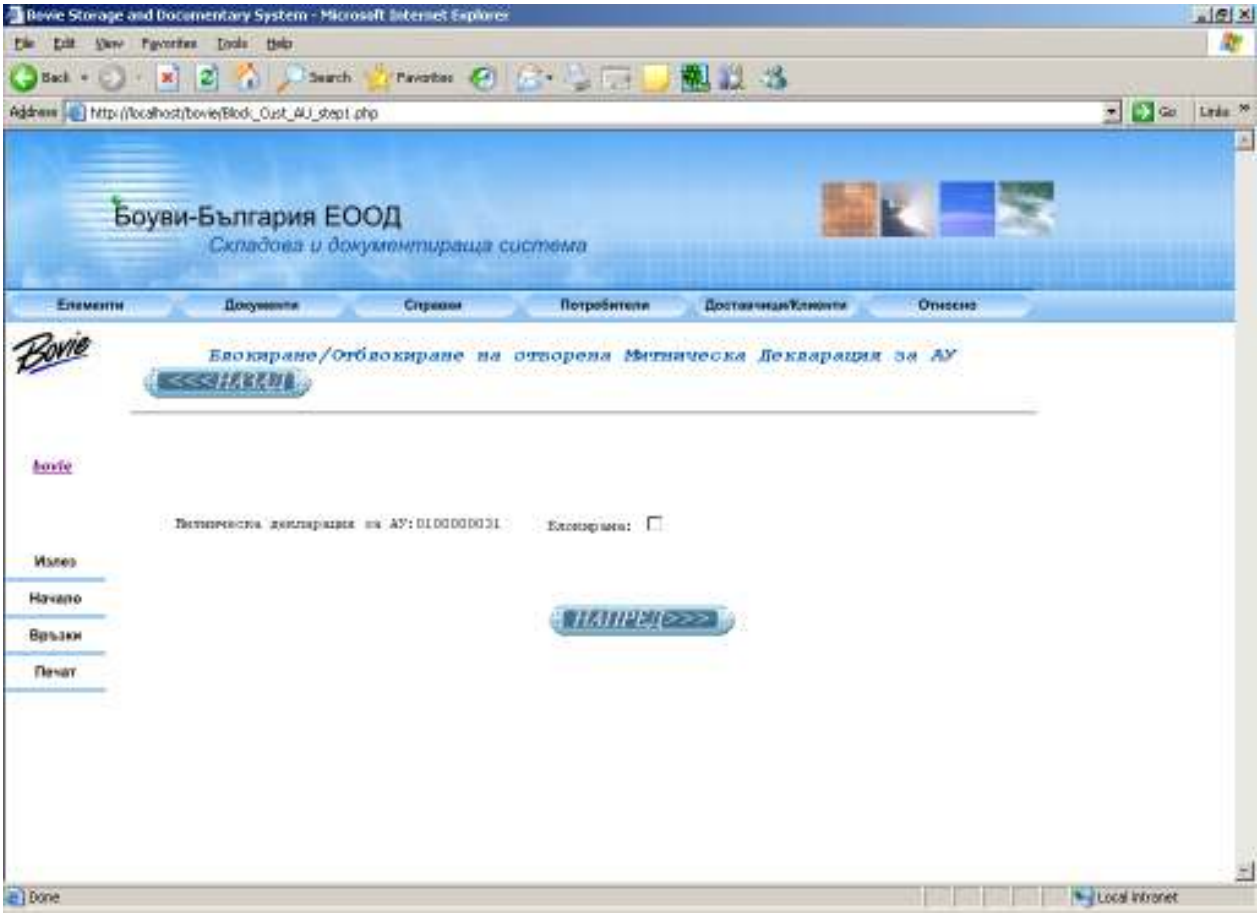
избраният компонент се зарежда в текстовото поле, а в полето за мерна единица се изписва неговата мерна единица.

Друг основен екран от системата е началния такъв при редактиране на документи.

За всички документи той е същия като изключим заглавието. Има поле за избор на документ за редактиране, позициите в което първоначално са сортирани по номер на документ (както се вижда по по-светлия бутон). Сортировката може да бъде променяна чрез другите бутони.

Ще завършим представянето на документите с изключение, каквото е възможността митническа декларация за активно усъвършенстване да бъде блокирана. Това от гледна точка на системата значи да се прекрати износа на компоненти внесени с нея. Това се постига като се промени статуса и.

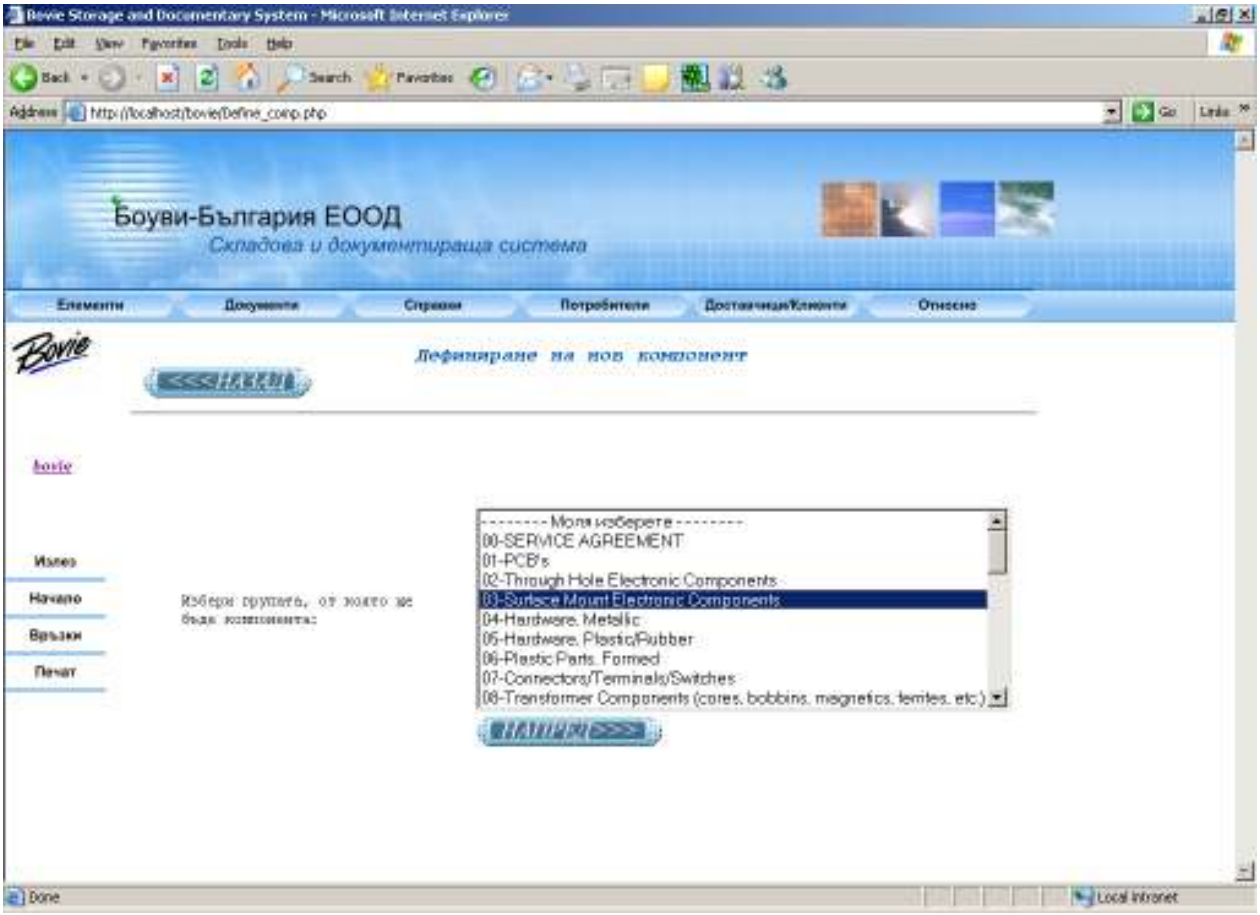
Избира се митническата, която искаме да блокираме или отблокираме и се натиска бутона НАПРЕД>>>.



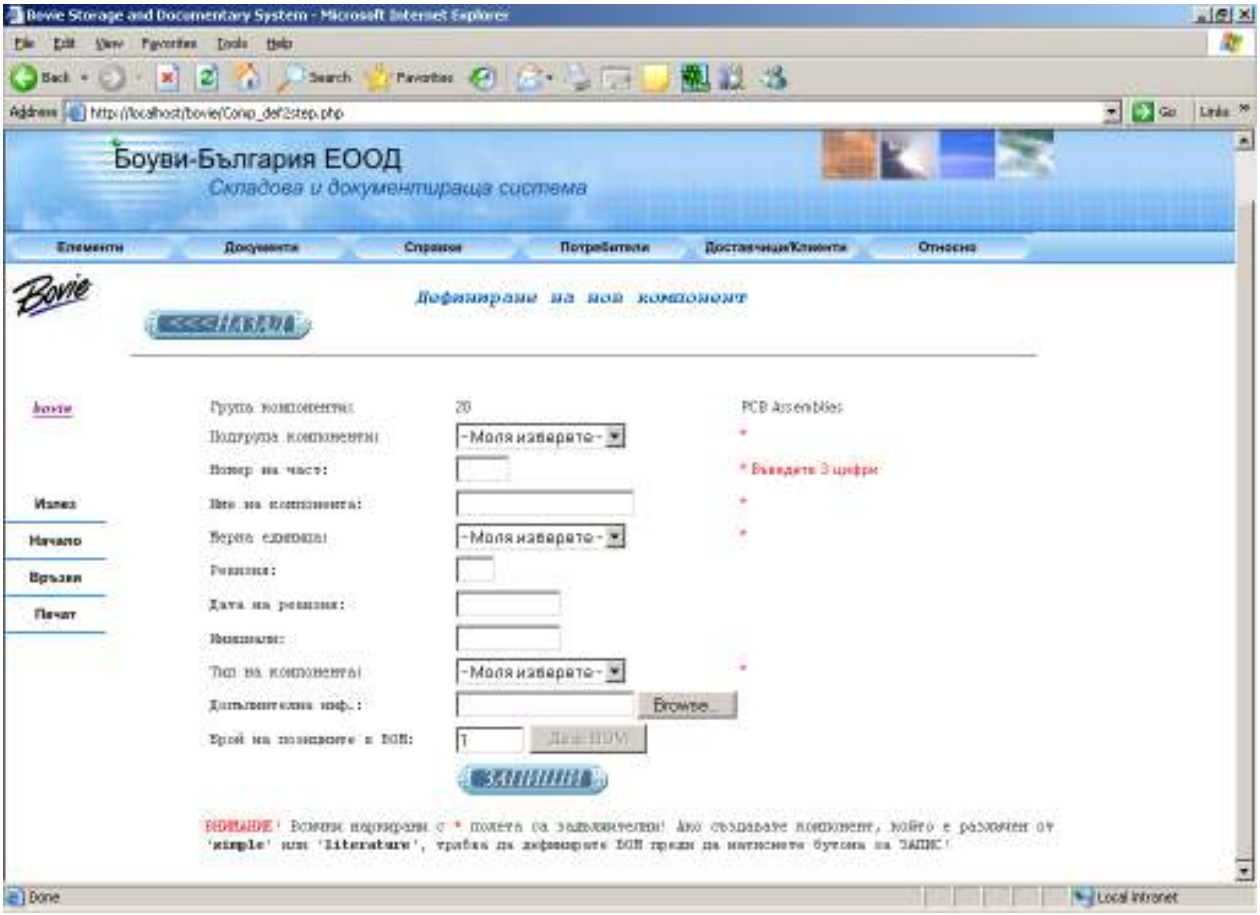
Ако има отметка в за блокиране при зареждане на този екран, значи тя е била блокирана до сега. В зависимост от отметката при натискане на НАПРЕД>>> избраната митническа се блокира или отблокира.

6.3. Основни записи на компоненти

При дефинирането на компонент се преминава през следните екрани:



Тук може да се види и друга форма на използване на елемента за избор. Видими са повече от един елемента. На екран, на който има място този начин на използване е подходящ. След натискане на  се преминава към следния екран:

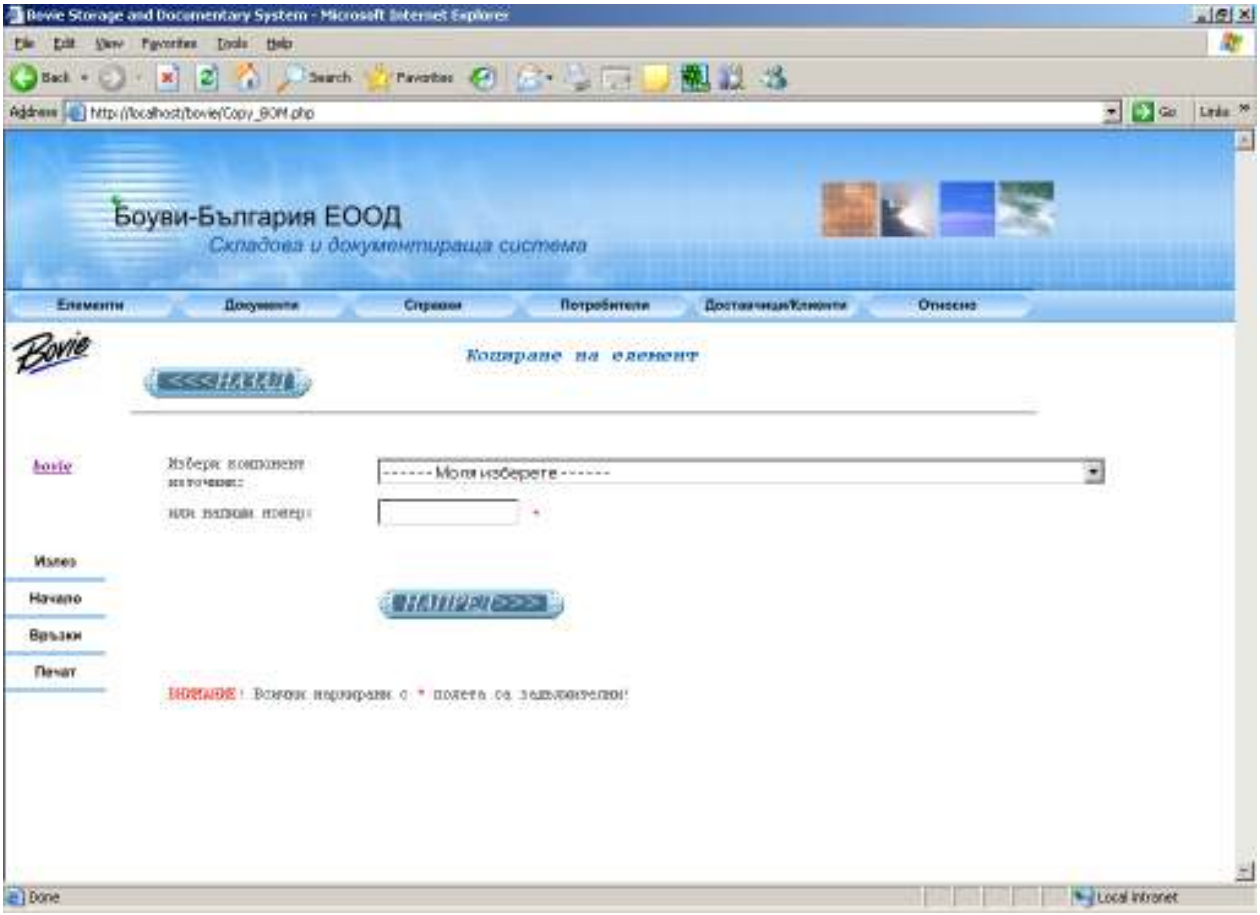


Тук особеното е че има бутон  , който се среща само на този екран и при редактиране на компонент. Чрез него може да бъде прикачен файл с характеристики или описание към основния запис на компонента. С натискането му се визуализира стандартна форма за избор на файл.

Заслужава да се спомене също, че бутона  е неактивен докато не се избере тип на компонент, от полето за избор, който е различен от ‘simple’ или ‘literature’.

Да се дефинира компонент е възможно и чрез функция за копиране от наличен такъв.

Първия екран, който се визуализира е следния:

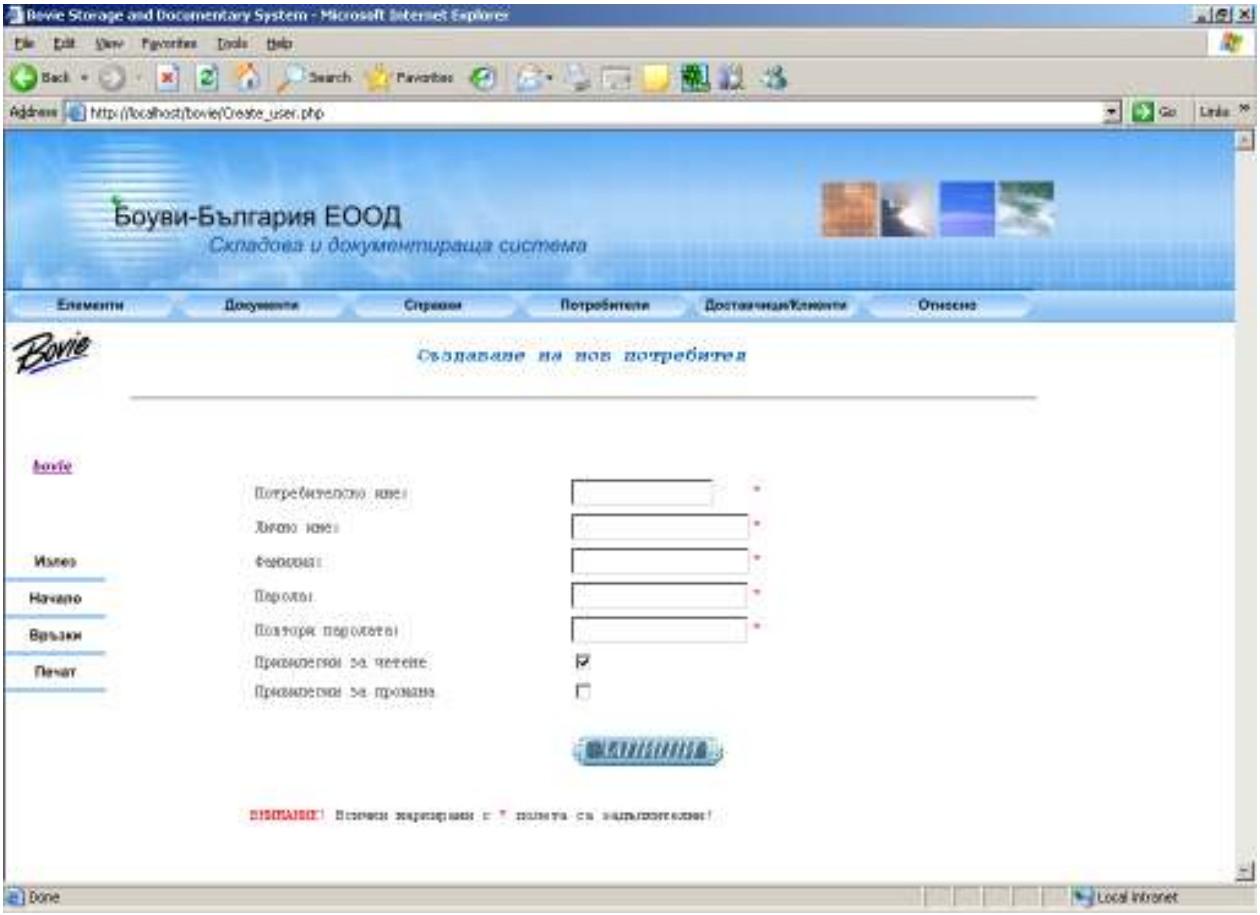


Още на много места в системата ще срещнем тази форма. В полето за избор са заредени всички компоненти и можем да изберем от тях. Дадена е и възможност да се използва , описаната в документи, функция ‘f, F’, за улесняване на търсенето. При избор на компонент, в текстовото поле се зарежда неговия номер. Възможно е също така директно да се напише номера в полето. В следващият екран се отварят форми със заредените характеристики на избрания компонент и отдолу неговите съставни ако има такива. Променят се желаните неща и новия компонент се записва. Този начин на създаване е особено полезен при дефиниране на нова ревизия на компонент.

Поради това, че интерфейсът е много по-опростен при дефинирането на групи и подгрупи няма да бъде разгледан тук.

6.4. Потребители и доставчици/клиенти

Само потребителят ‘bovie’ има право да работи с формите за потребители. Екрана за дефиниране е следния:



Както се вижда всички текстови полета са задължителни.

Ще се спрем само и на екрана за блокиране. Тези за редактиране и за изтриване са достатъчно ясни.

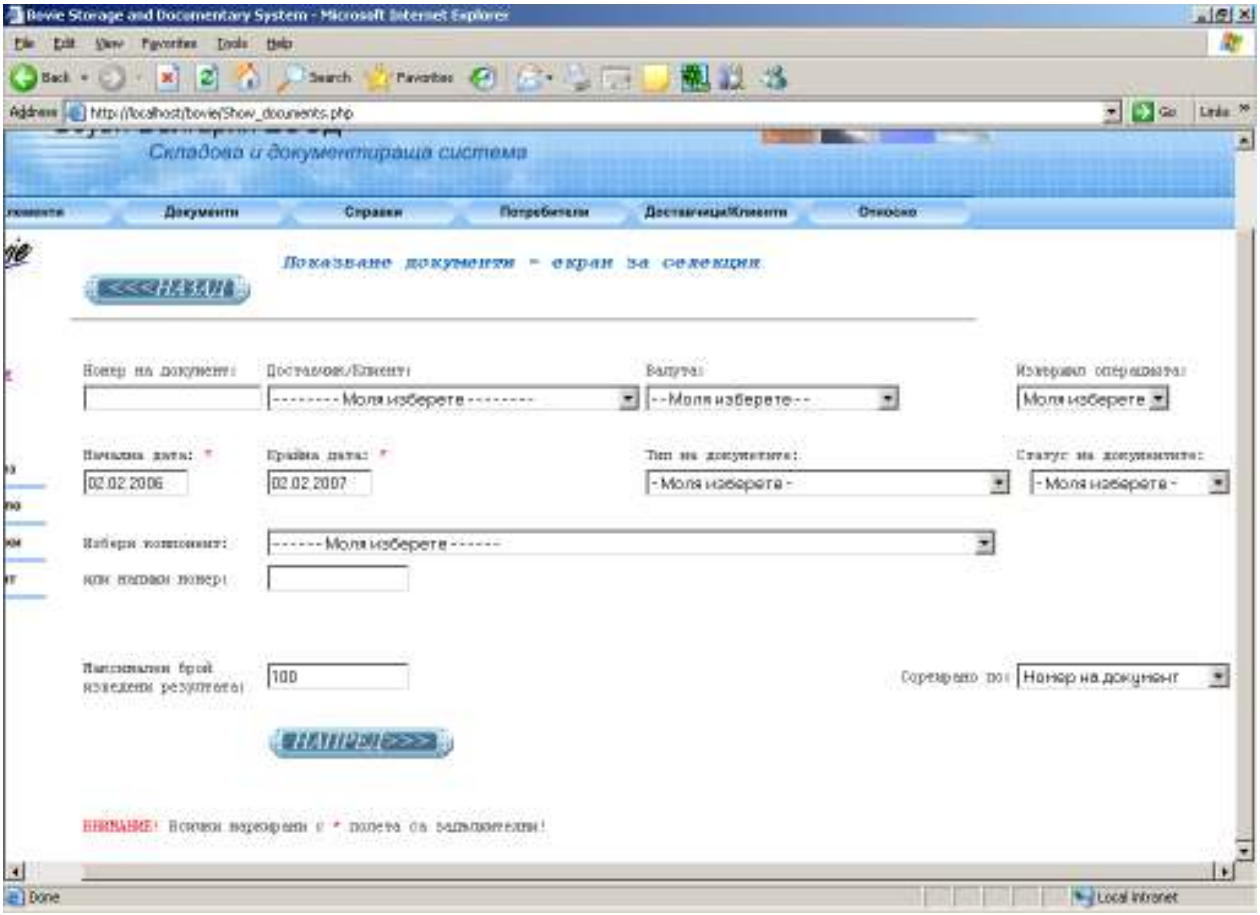
При зареждане на този екран потребителите, които са блокирани са с отметка. Нанасят се нужните промени и се натиска бутона БЛОКИРАЙ, който може би е по-правилно да бъде наречен ‘ПРОМЕНИ’.

При създаването на доставчици има подобни форми за създаване, редактиране и изтриване, както и списък с всички доставчици. Той има следния вид:

6.5. Справки

В системата има голямо количество справки и техния брой непрекъснато се увеличава. Ще бъдат разгледани част от тях за да се добие престава за интерфейса.

Следващият екран показва екрана за селекция на общата справка за документи. В нея могат да се видят всички налични данни за основния запис на документ и за неговите позиции.



От екрана се вижда по какви критерии могат да бъдат филтрирани документите. В досега разгледаните екрани никъде не споменахме за възможността да използваме символи за приближение при търсенето. Такъв може да се използва в полето номер на документ. Символа е ‘%’. Например ако искаме да намерим всички първи документи от всички типове, можем да напишем %1 в текстовото поле за номер на документ.

Друго характерно нещо, за екраните за селекция на справки, е това че обикновено в тях фигурират две полета за определяне на период от време, а не само едно за дата. Стойностите им по подразбиране обхващат период от една година назад, започвайки от днешната дата.

В горния екран има дадена възможност за избор на различни начини на сортиране, както и друго поле, което се среща не само тук, за специфициране на максималния брой върнати резултата. Неговата стойност по подразбиране е 100.

Примерен резултат от изпълнението на справката е следния:

Вижда се предупреждението, че има повече резултата, но е ограничено показването им. Номерът на документа представлява връзка към екран за редактиране на този документ. По този начин се улеснява потребителя при желание да нанесе корекции по даден документ. Същия подход е използван в част от другите справки, както за номера на документи, така и за номера на компоненти.

Друга справка, този път в табличен вид, е тази за движение на компонент

Движение на компонент

<<<НАЗАД

Компонент **PCB, 900-New-01-039-001** Мерна единица:**БР**
Селекция: Начална дата:**2001-02-02** Крайна дата:**2007-02-02**
Тип документи:**Всички**

Документ	Дата	Количество(БР)	Цена
МИТН.ДЕКЛАРАЦИЯ-АУ			
0100000016	13.05.2005	700	10.341996
0100000043	15.12.2005	800	11.45952
0100000058	19.04.2006	500	11.181726
0100000084	04.10.2006	750	10.646286
Сума МИТН.ДЕКЛАРАЦИЯ-АУ		2750	10.902760254545
ПРИЕМОПРЕДАВАТЕЛЕН ПРОТОКОЛ			
1100000001	16.05.2005	1	6.9
1100000051	08.12.2005	200	6.9
1100000070	05.05.2006	112	6.9
1100000075	18.05.2006	500	6.9
1100000077	20.01.2006	500	6.9
1100000080	24.02.2006	300	6.9
1100000104	06.11.2006	750	6.9
Сума ПРИЕМОПРЕДАВАТЕЛЕН		2363	6.9
ПРОТОКОЛ ЗА КАЧЕСТВО			
0900000884	01.09.2006	"700"	0
0900001277	01.09.2006	"800"	0
0900001583	25.04.2006	"500"	0
0900002192	11.10.2006	"750"	0
Сума ПРОТОКОЛ ЗА КАЧЕСТВО		0	0

Общо: Количество:387

При почти всички справки най-отгоре, под заглавната част, са изредени зададените критерии в екрана за селекция.

Тази справка дава движението за избран компонент по документи, т.е. показва всички документи в позициите, на които този компонент фигурира. В зависимост от типът им те внасят или изнасят от склада количества или нямат отношение към склада. Ако е обхванат целия период от време, за който има движения по документите за този компонент, общото количество накрая е същото както това в склада.

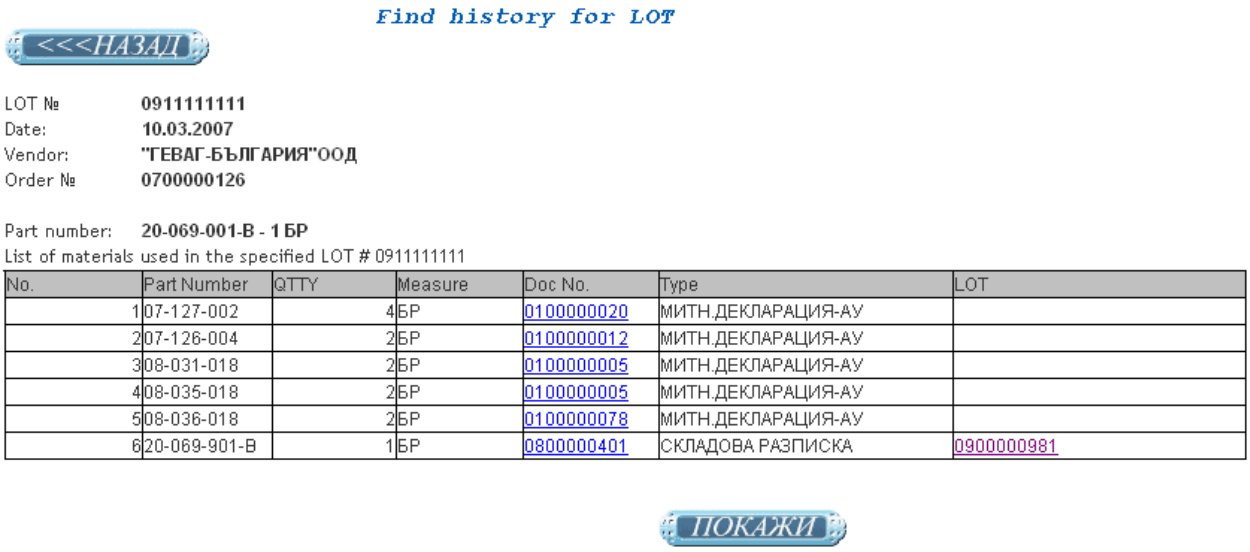
Друг вид таблици са тези по митническите декларации. Те са особено важни за представяне пред митническите власти и имат подобен вид:

Закриване по ЕАД 5100/5-1054/05.04.2005

No	Номенклатурен No	Описание на стоката	Мерна единица	Внесено количество	Остатък по документ	Изнесено с документ No/Дата	Изнесено количество
1	5156100002	Busswire, AWG20, BRB,	Ft.	2000	1637.6	0300000008/27.05.2005 121212/03.12.2006 123213/03.12.2006	348 6.4 8

За проследимостта на компонентите е направена справка. За да изведе очакваният резултат, е необходимо да се спазва стриктно попълването на документите с номерата на поръчките и ЛОТ номерата, както е описано в по-горните точки.

В екрана за селекция на справката се избира номер на протоколът за качество, с което Боуви е определила дадено изделие за годно. При проблем с него се тръгва от този протокол. След избирането на желания протокол се визуализира екран със съставните на окачественото изделие.



Find history for LOT

<<<НАЗАД

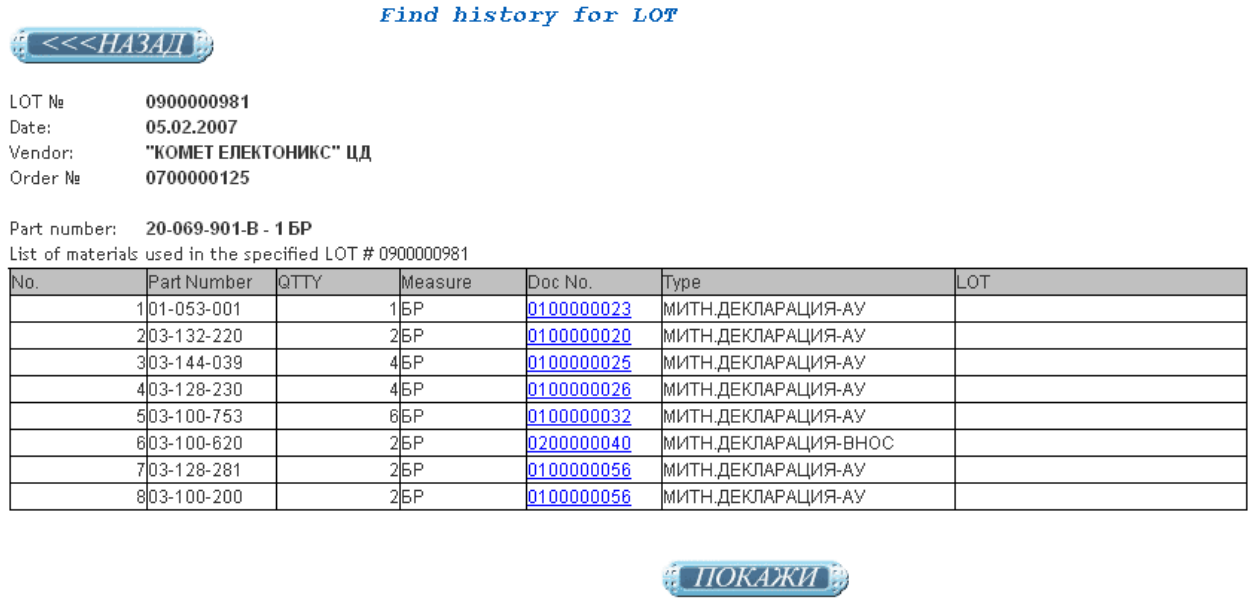
LOT № **0911111111**
Date: **10.03.2007**
Vendor: **"ГЕВАГ-БЪЛГАРИЯ"ООД**
Order № **0700000126**

Part number: **20-069-001-B - 1 БР**
List of materials used in the specified LOT # 0911111111

No.	Part Number	QTTY	Measure	Doc No.	Type	LOT
1	07-127-002	4	БР	0100000020	МИТН.ДЕКЛАРАЦИЯ-АУ	
2	07-126-004	2	БР	0100000012	МИТН.ДЕКЛАРАЦИЯ-АУ	
3	08-031-018	2	БР	0100000005	МИТН.ДЕКЛАРАЦИЯ-АУ	
4	08-035-018	2	БР	0100000005	МИТН.ДЕКЛАРАЦИЯ-АУ	
5	08-036-018	2	БР	0100000078	МИТН.ДЕКЛАРАЦИЯ-АУ	
6	20-069-901-B	1	БР	0800000401	СКЛАДОВА РАЗПИСКА	0900000981

ПОКАЖИ

При натискане на номерата в колона ‘Doc No.’ , се отваря съответния документ за редактиране. Ако ЛОТ номера е протокол за качество издаден от Боуви, тогава при избиране се отваря същата справка, но за съответния протокол за качество, в случая 0900000981



Find history for LOT

<<<НАЗАД

LOT № **0900000981**
Date: **05.02.2007**
Vendor: **"КОМЕТ ЕЛЕКТОНИКС" ЦД**
Order № **0700000125**

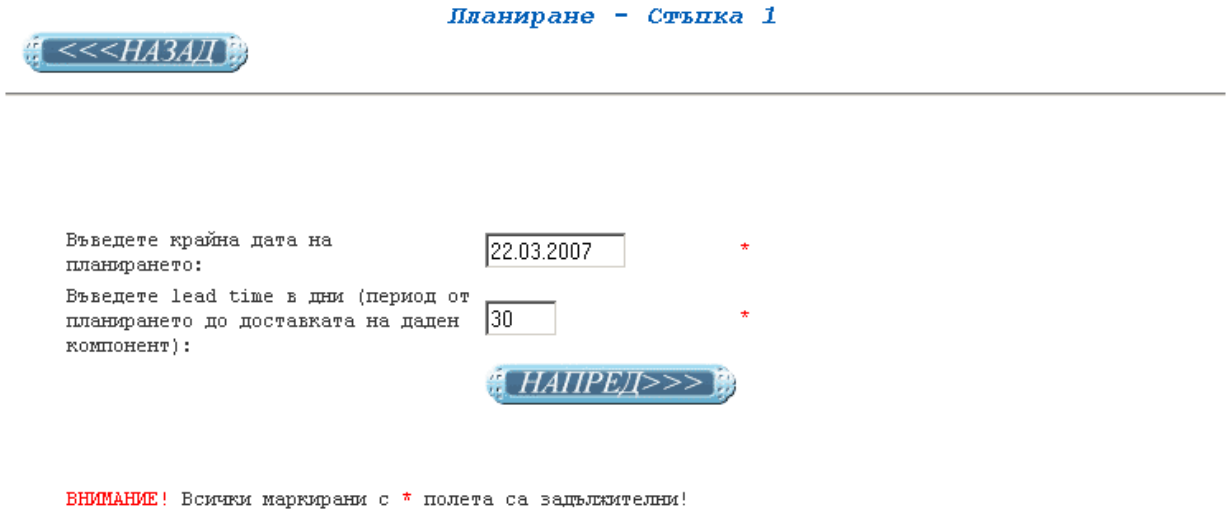
Part number: **20-069-901-B - 1 БР**
List of materials used in the specified LOT # 0900000981

No.	Part Number	QTTY	Measure	Doc No.	Type	LOT
1	01-053-001	1	БР	0100000023	МИТН.ДЕКЛАРАЦИЯ-АУ	
2	03-132-220	2	БР	0100000020	МИТН.ДЕКЛАРАЦИЯ-АУ	
3	03-144-039	4	БР	0100000025	МИТН.ДЕКЛАРАЦИЯ-АУ	
4	03-128-230	4	БР	0100000026	МИТН.ДЕКЛАРАЦИЯ-АУ	
5	03-100-753	6	БР	0100000032	МИТН.ДЕКЛАРАЦИЯ-АУ	
6	03-100-620	2	БР	0200000040	МИТН.ДЕКЛАРАЦИЯ-ВНОС	
7	03-128-281	2	БР	0100000056	МИТН.ДЕКЛАРАЦИЯ-АУ	
8	03-100-200	2	БР	0100000056	МИТН.ДЕКЛАРАЦИЯ-АУ	

ПОКАЖИ

Последната справка, която ще бъде разгледана е тази за планирането. Тя е най-важната за полезността на системата за фирмата, защото дава възможност за автоматизиране и улесняване на процеса по планиране на доставките на компоненти. Ясно е от каква важност е този процес, за фирма занимаваща се с производство.

Първата стъпка е да се зададе дата за планиране и времето необходимо за производство.



След натискане на бутона за продължение се визуализира следния екран:

Необходими компоненти за поръчка към дата 22.03.2007

<<<НАЗАД

Проформа: **0600000437/28.01.2007** Планирай

1. *02.02.2007* *nov unit 1(99-999-501), 50 LB*

	Номенклатурен No	Наименование на стоката	Количество
☐	99-999-501	nov unit 1	50
☐	99-999-402	nov module 2	50
☐	99-999-101	nov simple 1	14
☐	99-999-102	nov simple 2	100
☐	99-999-103	nov simple 3	150
☐	99-999-401	nov module 1	97
☐	99-999-201	nov subassembly 1	91
☐	99-999-101	nov simple 1	93
☐	99-999-102	nov simple 2	186
☐	99-999-301	nov assembly 1	190
☐	99-999-101	nov simple 1	190
☐	99-999-201	nov subassembly 1	380
☐	99-999-101	nov simple 1	380
☐	99-999-102	nov simple 2	760
☐	99-999-202	nov subassembly 2	560
☐	99-999-101	nov simple 1	560
☐	99-999-102	nov simple 2	1120

Екрана е разделен по проформи, т.е. планирането става за отделна проформа. От тук трябва да се зададат компонентите, които искаме да поръчаме (количествата в последната колона са необходимите такива, т.е. ако сме имали наличности в склада или поръчани вече компоненти, те са извадени от цялото необходимо количество).

След натискането на бутона за планиране на конкретна проформа се отваря нов прозорец.

Необходими компоненти за поръчване за проформа 0600000437/28.01.2007

<<<НАЗАД

Доставчик: "СТЕДИ-ЕЛ" ООД

Генерирай

No	Номенклатурен No	Наименование на стоката	Количество	
1	99-999-101	nov simple 1	1237	03.01.2007
2	99-999-102	nov simple 2	2166	03.01.2007
3	99-999-103	nov simple 3	1830	03.01.2007

Доставчик: "АМЕТА"ООД

Генерирай

No	Номенклатурен No	Наименование на стоката	Количество	
1	99-999-401	nov module 1	97	03.01.2007

Доставчик: ЕЛТЕХ ООД

Генерирай

No	Номенклатурен No	Наименование на стоката	Количество	
1	99-999-201	nov subassembly 1	471	03.01.2007
2	99-999-301	nov assembly 1	190	03.01.2007

В него вече са сумирани еднаквите компоненти за поръчване и са групирани по дати и по доставчици. По доставчици са групирани по признак, кой е последния такъв, който е доставял от съответния компонент.

Натискането на бутон Генерирай отваря стандартна поръчка за доставка с попълнени всички полета с изключение на цената.

Гореописаният процес веднъж показва автоматично какво е необходимо да се поръча, на базата на отворените проформи, поръчки и складови наличности и второ подпомага процеса по поръчването им.

7. Изводи и заключения

Както беше обяснено системата е разработена конкретно за нуждите на „Боуви-България” ЕООД. Тя е насочена към специфичните изисквания на фирмата и е въведена успешно в експлоатация.

При направеното сравнение с модули предлагащи подобна функционалност в SAP се видя, че откъм предлагани възможности и гъвкавост складовата и документираща система, тема на тази дипломна работа, отстъпва, но не сравнението с водещи в областта продукти е нейна цел, а задоволяване потребностите на „Боуви-България” ЕООД . Използването на изцяло свободно разпространявани решения при нея я прави несравнимо по-евтина и по-лесна за поддържане и разширяване.

За разработката и са използвани WEB технологии, които с добрите канали за достъп до Интернет в днешно време, я правят достъпна от целия свят. Това разбира се не е само положително, но носи и отговорности по осигуряване на защитата й от неоторизиран достъп. На това е обърнато внимание не само при разработването на системата, но и при вкарването и в експлоатация. Чрез защитна стена се пропускат само определени IP адреси за достъп до нея.

За бъдещото развитие на системата може да се каже, че още при избора на технологии е залегнало правилото, че това ще бъде динамично променяща се система, която ще може да следва непрекъснато променящите се нормативни актове и закони, които променят и външния вид и характеристиките на някои от документите. Най пресен пример за това са промените по фактурите от началото на 2007-ма година. Разделянето на реализацията на голям брой файлове, за различните страници от приложението, благоприятства сравнително лесно да се правят промени по външния вид и да се променят характеристиките на дадена функционалност, без да се рискува създаването на проблеми в друга функционалност. Това спестява и време за тестване при такива промени.

За най-близко бъдеще е планирано реализирането на функционалност за архивиране, както и по-усложнена такава за определяна на потребителските права, а за малко по-далечно бъдеще ще се създадат и възможности за експорт на различни типове документи.

Предвижда се функционалността за архивиране, да прехвърля документите от текущите таблици към подобни такива, за интервал от време в миналото, който още не е уточнен, а след това да има възможност при справките, за които се сметне че е необходимо, да се даде възможност на потребителите да избират дали в разреза от базата данни, който им определя дадена справка да се включат или не архивираните вече документи.

В заключение може да се каже, че сравнително удобния интерфейс помага за лесното и усвояване и експлоатиране от страна на потребителите, които я определят като полезна, което е най-важният резултат от създаването на какъвто и да е софтуер.

8. Използвани източници

1. http://www.sap.com/
2. Софтуерни технологии – Аврам Ескенази, Нели Манева, КЛМН, София, 2006
3. PHP Manual - Edited by Stig Sæther Bakken, Egon Schmid
4. MySQL 5.0 Reference Manual
5. JavaScript Application Cookbook By Jerry Bradenbaugh, O'Reilly, Sept 1999
6. PHP Cookbook By David Sklar, Adam Trachtenberg, O'Reilly, November 2002
7. MySQL Cookbook By Paul DuBois, O'Reilly, October 2002

9. Приложения

Функция FilterComponents

```
function FilterComponents()
{
	if(
		(
			(navigator.appName == "Microsoft Internet Explorer")	&&
			((event.keyCode == 70) || (event.keyCode == 102)) // "f" ili "F"
		)
	  )
	{
		var Str = window.prompt("Въведете част от името или номера на търсения
		компонент:", "");
		if(Str != null)
		{
			var Index;

			with(document.forms.Choosecomponent.id_components)
			{
				for(Index = 0; Index < options.length; Index++)
				{
					if(options[Index].id.substring(0, Str.length) == Str)
					{
						selectedIndex = Index;
						document.forms.Choosecomponent.id_components_t
						ext.value = options[Index].id;
						break;
					}
				}
				if(Index == options.length)
				{
					selectedIndex = 0;
					document.forms.Choosecomponent.id_components_text.val
					ue = "";
				}
			}
		}
	}
}
```

Запис на файл на сървъра при създаване на компонент

```
if ($_FILES['PDF']['tmp_name']<>""){
                        $data = addslashes("/files/".$id_part_rev."_".$_FILES['PDF']['name']);
                         }
                         else {
                                $data="";
                         }


                        if (($comp_type=='1') or ($comp_type=='6'))
                        {
                                $query="insert into components values
                                ('','$id_part_rev','$part_num','$comp_name','$comp_type','$id_meas
                                ure','$data','$part_rev','$data_revision','$initials','$id_grp','$id_sgrp','
                                ','');";
                                $result=mysql_query($query, $conn);
                                if ($result=='1')
                                {
                                        if ($data<>"") {
                                                //Copy the file to some permanent location
                                                if(!move_uploaded_file($_FILES["PDF"]["tmp_nam
                                                e"], "/home/www/intrsk".$data))
                                                  {
                                                        $msg="Има проблем със записа на
                                                        допълнителният файл!!!";
                                                  }
                                          }

                                        $msg=$msg."Компонентът '".$comp_name."' е създаден!";
                                }
                                else
                                {
                                        $msg = ErrorMessage();
                                }
                        }
```

Проверка коректността на html форма с JavaScript

Примера е от дефинирането на митническа за активно усъвършенстване.

```
//дали е валиден номер на компонент
function IsValidPartRev(Str, I)
{
        var Index;
        var Flag = false;
        for(Index in ArrayComp)
        {
                if((ArrayComp[Index] == Str) && (ArrayComp[Index] == Str))
```

```
                {
                        eval("document.forms.def_cust_AU.id_components" + I).value = Index;
                        Flag = true;
                        break;
                }
        }

        return Flag;
}


//проверка на валутата
function CheckCurrency()
{
        var SelectedI = document.forms.def_cust_AU.id_currency.selectedIndex;
        if((document.forms.def_cust_AU.id_currency.options[SelectedI].value == 1) &&
(document.forms.def_cust_AU.curr_rate_to_bgn != 1))
        {
                document.forms.def_cust_AU.curr_rate_to_bgn.value = 1;
                alert("Избраната валута е BGN! Валутният курс трябва да бъде 1!");
        }
}


// дали количествата са коректни
function IsQuantitiesOk()
{
        var Index;
        for(Index = 1; Index <= <?php echo($position_number); ?>; Index++)
        {
                with(eval("document.forms.def_cust_AU.quantity" + Index))
                {
                        if((IsFloat(value) == false)  || (value == ""))
                        {
                                alert("Въведете коректно количество XXXXXXXX!");
                                focus();
                                return false;
                        }
                }
        }
}


// дали е избран компонент
function IsCompOk()
{
        var Index;
```

```
	for(Index = 1; Index <= <?php echo($position_number); ?>; Index++)
	{
		with(eval("document.forms.def_cust_AU.id_components" + Index))
		{
			If ( (value == "")	&&
			(IsValidPartRev(eval("document.forms.def_cust_AU.comp" +
			Index).value, Index) == false)  )
			{
				alert("Изберете компонент или въведете валиден номер на
				компонент!");
				(eval("document.forms.def_cust_AU.comp" + Index)).focus();
				return false;
			}
		}
	}
}

//дали е избран доставчик
function IsVendorOk()
{
	with(document.forms.def_cust_AU.id_vc)
	{
		if(value == "")
		{
			alert("Изберете доставчик!");
			focus();
			return false;
		}
	}
}

// дали е избран компонент
function IsPriceOk()
{
	var Index;
	for(Index = 1; Index <= <?php echo($position_number); ?>; Index++)
	{
		with(eval("document.forms.def_cust_AU.price" + Index))
		{
			if((IsFloat(value) == false) || (value == ""))
			{
				alert("Въведете коректна цена XXXX.XXXX!");
				focus();
				return false;
			}
		}
	}
```

```
}

// дали валутния курс е ОК
function IsRateOk()
{
	with(document.forms.def_cust_AU.curr_rate_to_bgn)
	{
			if((IsFloat(value) == false) || (value == ""))
			{
				alert("Въведете коректен курс!");
				focus();
				return false;
			}
	}
}

function IsDateOk()
{
	with(document.forms.def_cust_AU.data_of_doc)
	{
		if(CheckInputDate(value) == false)
		{
			alert("Въведете датата във формат: DD.MM.YYYY");
			focus();
			return false;
		}
	}
}
```

Транзакции в базата от данни

Примера е от дефиниране на митническа декларация за активно усъвършенстване

```
//старт на транзакция
mysql_query('SET AUTOCOMMIT = 0');
mysql_query('BEGIN WORK');

$msg = CreateDocument(1, $id_vc, 5, $data_of_doc, $position_number, $id_currency,
$curr_rate_to_bgn, 1, $reference_1, $reference_2, $arr);

// input in storage
if($msg == "")
{
	$msg = InputInStorage($arr, 5);
	// manege the status of the order docs
	if($msg == "")
	{
```

```
                $msg = ManageDocStatusPor($arr, $id_vc);
        }
}

// край на транзакция
if($msg == "")
{
        $msg = "Митническата декларация ".$num_doc." е въведена и складовите
        наличности са променени.";
        mysql_query('COMMIT WORK');
}
else
{
        mysql_query('ROLLBACK WORK');
}
mysql_query('SET AUTOCOMMIT = 1');
```

Функция CreateDocument

Чрез нея се създават всички документи с техните позиции, с изключение на протоколите за качество и на фактурите.

```
function CreateDocument($id_type_doc, $id_vc, $id_action, $data_of_doc, $position_number,
$id_currency, $curr_rate_to_bgn, $id_stat_doc, $reference1, $reference2, $doc_positions_array)
{
        //осигурява връзката с базата данни
        require("Connect.php");

        // превръща датата във формат за запис в базата данни
        $db_data_of_doc = convert_date_n_to_db($data_of_doc);
        $db_data_of_operation = date("Y-m-d");

        // get ID_USER
        $query = "SELECT ID_USER FROM users WHERE USER_NAME =
        '".$_SESSION['username']."';";
        $result = mysql_query($query, $conn) or
                die("<h1>Database query couldn't be executed.<br>Probably MySQL server is not
                started.</h1>");
        $row_user = mysql_fetch_array($result);
        $id_user = $row_user['ID_USER'];

        // get NUM_DOC
        $num_doc = ReturnNumDoc($id_type_doc);

        // create new document in doc_recs
```

```
$query = "INSERT INTO doc_recs VALUES ('','$num_doc', '$id_type_doc', '$id_vc',
'$id_action', '$id_user', '$db_data_of_doc', '$db_data_of_operation', '$position_number',
'$id_currency', '$curr_rate_to_bgn', '$id_stat_doc', '$reference1', '$reference2');";
$result_doc_recs = mysql_query($query, $conn);
$msg = "";
if($result_doc_recs == 0)
{
        $msg = "Грешка при създаването на документ в таблицата doc_recs!
        <br>".ErrorMessage();
}
else
{
        // create document positions
        $query = "SELECT MAX(ID_DOC) from doc_recs;";
        $result = mysql_query($query, $conn) or
                die("<h1>Database query couldn't be executed.<br>Probably MySQL
                server is not started.</h1>");
        $row = mysql_fetch_array($result);
        $id_doc = $row['MAX(ID_DOC)'];

        foreach($doc_positions_array as $arr_var)
        {
                $query = "INSERT INTO doc_pozitions VALUES ('', '".$id_doc."',
                '".$arr_var['quantity']."', '".$arr_var['id_components']."',
                '".$arr_var['price']."', '".convert_date_n_to_db($arr_var['dead_line'])."',
                '".$arr_var['ref_doc_num']."', '".$arr_var['num_ead']."',
                '".$arr_var['quantity']."', '".$arr_var['lot_number']."');";
                $result_doc_positions = mysql_query($query, $conn);
                if($result_doc_positions == 0)
                {
                        $msg = $msg."<br> Грешка при създаването на запис в
                        таблицата doc_pozitions! <br>".ErrorMessage();
                        break;
                }
        }

        // increment count_doc_type (the count of the documents from the selected type)
        $query = "UPDATE docs_type SET COUNT_TYPE_DOC =
        COUNT_TYPE_DOC + 1 WHERE ID_TYPE_DOC = '$id_type_doc';";
        $result = mysql_query($query, $conn);
        if($result == 0)
        {
                $msg = $msg."<br> Грешка при обновяването на запис в таблица
                docs_type! <br>".ErrorMessage();
        }
}
return $msg;
```

```
}
```

Функция InputInStorage

Въвежда в склада компонентите от параметъра си масив

```
function InputInStorage($arr, $id_action)
{
	require("Connect.php");

	$msg = "";
	foreach($arr as $arr_var)
	{
		$query = "SELECT * FROM storage WHERE ID_COMPONENTS =
		".$arr_var['id_components'].";";
		$result = mysql_query($query, $conn) or
			die("<h1>Database query couldn't be executed.<br>Probably MySQL
			server is not started.</h1>");
		$row = mysql_fetch_array($result);
		if($row != 0)
		{
			$query = "UPDATE storage SET REAL_QUANTITY =
			REAL_QUANTITY + ".$arr_var['quantity']." WHERE
			ID_COMPONENTS = ".$arr_var['id_components'].";";

			$result = mysql_query($query, $conn);
			if($result == 0)
			{
				$msg = $msg."<br> В склада не е обновено количеството на
			компонент с ID = .$arr_var['id_components'].".<br>" .ErrorMessage();
			}
		}
		else
		{
			$query = "INSERT INTO storage VALUES (", '".$arr_var['quantity']."',
			'".$arr_var['id_components']."');";

			$result = mysql_query($query, $conn);
			if($result == 0)
			{
				$msg = $msg."<br> В склада не е внесен компонент с  ID =
				".$arr_var['id_components'].".<br>".ErrorMessage();
			}
		}
	}

	return $msg;
```

```
}
```

Функция OutputFromStorage

Изважда от склада количествата на компонентите подадени в параметъра масив.

```
function OutputFromStorage($arr)
{
	require("Connect.php");

	// sumira ednakvite elementi
	$arr_new = array();
	$arr_count = count($arr);
	for($i = 0; $i < $arr_count; $i++)
	{
		$arr_new_count = count($arr_new);
		for($j = 0; $j < $arr_new_count; $j++)
		{
			if($arr[$i]['id_components'] == $arr_new[$j]['id_components'])
			{
				$arr_new[$j]['quantity'] = $arr_new[$j]['quantity'] +
				$arr[$i]['quantity'];
				break;
			}
		}

		if($j == $arr_new_count)
		{
			array_push($arr_new, $arr[$i]);
		}
	}

	$arr = $arr_new;


	// proveriava dali kolichestvata sa dostatychni v sklada za da moje da se iznese
	foreach($arr as $arr_var)
	{
		if(ReturnComponentAvailability($arr_var['id_components']) <
		$arr_var['quantity'])
		{
			return $arr_var['id_components'];
		}
	}

	foreach($arr as $arr_var)
	{
```

```
		$query = "UPDATE storage SET REAL_QUANTITY = REAL_QUANTITY -
		".$arr_var['quantity']." WHERE ID_COMPONENTS =
		".$arr_var['id_components'].";";

		$result = mysql_query($query, $conn);
		if($result == 0)
		{
			return $arr_var['id_components'];
		}
	}

	return 0;
}
```

Функция ManageDocStatusPor

Променя статуса на поръчките. Използва се при складовите разписки, митническите декларации за редовен внос и за активно усъвършенстване.

```
function ManageDocStatusPor($arr, $id_vc)
{
	require("Connect.php");

	$msg = "";
	foreach($arr as $arr_var)
	{
		$res_por_arr = array();

		// izbira poziciite ot porychkata
		$query = "SELECT doc_pozitions.ID_DOC, SUM(QUANTITY),
		doc_pozitions.ID_COMPONENTS FROM doc_pozitions, doc_recs WHERE
		(doc_pozitions.ID_DOC = doc_recs.ID_DOC AND doc_recs.NUM_DOC  =
		'".$arr_var['ref_doc_num']."') GROUP BY doc_pozitions.ID_DOC,
		doc_pozitions.ID_COMPONENTS;";

		$result = mysql_query($query, $conn) or
			die("<h1>Database query couldn't be executed.<br>Probably MySQL
			server is not started.</h1>");
		while($row = mysql_fetch_array($result))
		{
			// $res = 3 -> tozi komponent vse oshte ne e dostaven
			// $res = 2 -> tozi komponent e dostaven no v po-malki ot porychanite
			//kolichestva
			// $res = 1 -> tozi komponent e dostaven v porychanite kolichetsva ili
			//poveche

			$res = 3;
```

```
		// izbira sumata na kolichestvata ot edin komponent ot poryckata v  SR
		$query_comp = "SELECT SUM(QUANTITY) FROM doc_pozitions,
			doc_recs WHERE (doc_pozitions.ID_DOC = doc_recs.ID_DOC
			AND doc_pozitions.REF_DOC_NUM  =
			'".$arr_var['ref_doc_num']."' AND (doc_recs.ID_TYPE_DOC  = '1'
			or doc_recs.ID_TYPE_DOC  = '2' or doc_recs.ID_TYPE_DOC  =
			'8') and doc_pozitions.ID_COMPONENTS  =
			'".$row['ID_COMPONENTS']."' and doc_recs.ID_VC =
			'".$id_vc."');";

			$result_comp = mysql_query($query_comp, $conn) or
				die("<h1>Database query couldn't be
				executed.<br>Probably MySQL server is not
				started.</h1>");
			$row_comp = mysql_fetch_array($result_comp);

			if(($row_comp['SUM(QUANTITY)'] != "") and
			($row_comp['SUM(QUANTITY)'] > 0))
			{
				// dali komponenta e dostaven vyv porychanite kolichetsva
				if($row_comp['SUM(QUANTITY)'] >=
				$row['SUM(QUANTITY)'])
			{
				$res = 1;
			}
			else
			{
				$res = 2;
			}
		}

		array_push($res_por_arr, $res);
}

// porychkata se zakriva chastichno
$min_res = GetMinElemInArray($res_por_arr);
$max_res = GetMaxElemInArray($res_por_arr);

// porychkata se zatvaria
if(($min_res == 1) and ($max_res == 1))
{
		$query = "UPDATE doc_recs SET ID_STAT_DOC = '2' WHERE
		NUM_DOC = '".$arr_var['ref_doc_num']."';";
		$result = mysql_query($query, $conn);

		if($result == 0)
```

```
                {
                        $msg = "Грешка при обновяване на статуса на документ";
                        break;
                }
        }
        else
        {
                $query = "UPDATE doc_recs SET ID_STAT_DOC = '1' WHERE
                NUM_DOC = '".$arr_var['ref_doc_num']."';";

                $result = mysql_query($query, $conn);

                if($result == 0)
                {
                        $msg = " Грешка при обновяване на статуса на документ";
                        break;
                }
        }
    }
    return $msg;
}
```

Функция CreateRefPositions

Прави запис в таблицата, в която се пазят данни за документите реферираци към позиция на документа

```
function CreateRefPositions($doc_positions_array)
{
    require("Connect.php");

    $query = "SELECT MAX(ID_DOC) from doc_recs;";
    $result = mysql_query($query, $conn) or
        die("<h1>Database query couldn't be executed.<br>Probably MySQL server is not
        started.</h1>");
    $row = mysql_fetch_array($result);
    $id_doc = $row['MAX(ID_DOC)'];

    foreach($doc_positions_array as $arr_var)
    {
        $query = "INSERT INTO doc_pozitions_ref VALUES (", '".$id_doc."',
        '".$arr_var['quantity']."', '".$arr_var['id_components']."',
        '".$arr_var['ref_doc_num']."');";

        $result_doc_positions = mysql_query($query, $conn);
        if($result_doc_positions == 0)
        {
```

```
                $msg = $msg."<br> Грешка при създаването на запис в таблицата
                doc_pozitions_ref! <br>".ErrorMessage();
            }
        }

        return $msg;
}
```

Функция GetComponentPart

Дефиниране на масив от всички подкомпоненти, от които се състои даден компонент. Данните, които се връщат са: позиция, идентификатор, наименование, количество, вид, дълбочина.

```
function GetComponentPart($id_comp, $index)
{
        require("Connect.php");
        $query = "select bom_definition.ID_COMPONENTS, PART_BOM, QUANTTY_BOM
        from components, bom_definition where
        ((components.ID_BOM=bom_definition.ID_BOM)and(components.ID_COMPONENTS
        ='$id_comp')) order by PART_BOM;";

        $result = mysql_query($query, $conn) or
                die("<h1>Database query couldn't be executed.<br>Probably MySQL server is not
                started.</h1>");

        $arr = array();

        while($row=mysql_fetch_array($result))
        {
                $id_components=$row['ID_COMPONENTS'];

                $query="select * from components, type_of_comp, measures where
                ((components.ID_COMPONENTS='$id_components') and
                (components.id_type_comp=type_of_comp.id_type_comp) and
                (components.ID_MEASURE=measures.ID_MEASURE));";

                $result_comp = mysql_query($query, $conn) or
                        die("<h1>Database query couldn't be executed.<br>Probably MySQL
                        server is not started.</h1>");
                $row_comp = mysql_fetch_array($result_comp);

                $arr_comp = array('position' => $row['PART_BOM'], 'part_id_rev' =>
                $row_comp['ID_PART_REV'], 'comp_name' => $row_comp['COMP_NAME'],
                'quantity' => $row['QUANTTY_BOM'], 'type_comp_name' =>
                $row_comp['TYPE_COMP_NAME'], 'depth' => $index, 'id_components' =>
```

```
                $row_comp['ID_COMPONENTS'], 'measure' =>
                $row_comp['MEASURE_SHORT_NAME']);

                array_push($arr, $arr_comp);
                $arr = array_merge($arr, GetComponentPart($id_components, $index + 1));
        }

        return $arr;
}
```