**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД**

**«ПРИАЗОВСЬКИЙ ДЕРЖАВНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ»**

**КАФЕДРА КОМП'ЮТЕРНИХ НАУКИ**

**ПОЯСНЮВАЛЬНА ЗАПИСКА**

**до курсової роботи**

**за темою «Розробити автоматизовану інформаційну систему обліку роботи стоматологічного центру»**

**Виконав:**

**студент групи КН-12**

**Романов К.Г.**

**Перевірив:**

**доцент, к.т.н.**

**Мироненко Д.С.**

**П`ятикоп Э.Э.**

**Маріуполь, 2015**

**Реферат**

Темою даного дипломного проекту є розробка автоматизованої інформаційної системи обліку роботи стоматологічного центру «АВАС-ДЕНТ».

Головні завдання, поставлені при написанні дипломного проекту - це аналіз предметної області та маркетингове дослідження, розробка постановки завдання для обліку роботи стоматологічного центру «АВАС-ДЕНТ», побудова UML-діаграм, розробка програмного забезпечення та супровідної документації, а також аналіз отриманих результатів, побудова діаграм різних методології, а також аналіз отриманих результатів.

Пояснювальна записка складається з таких основних частин, як загальний розділ, спеціальний розділ.

Введення дозволяє вивчити основні функції та призначення розроблюваної програми.

Спеціальна частина дипломного проекту включає аналіз предметної області - опис завдання і маркетингове дослідження, постановку задачі, моделювання об'єкта проектування і програмне забезпечення.

В аналізі предметної області описується сама область та обгрунтування необхідності розробки програмного забезпечення, також проводиться маркетингове дослідження аналогічних програмних продуктів.

У постановці завдання описується найменування програми, призначення та область її застосування, описані вимоги до функціональних характеристик програми, надійності програми, кваліфікації та чисельності персоналу, технології зберігання та обробки інформації проектованої завдання, СУБД, технічних і програмних засобів, вихідних кодів і мов програмування, організації вхідних і вихідних даних, захисту інформації та програм. Крім того, описаний попередній склад програмної документації.

Моделювання об'єкта проектування представлено основними UML-діаграмами, які дають картину використання, функціонування, компонентів і структури програмного забезпечення, основними діаграмами, які дають картину структури бізнес-процесів, їх взаємодії при проектуванні системи.

У програмному забезпеченні дано обґрунтування вибору СУБД, мови програмування та інструментальних можливостей програмної реалізації віртуальної лабораторії, інструкція користувача.

У переліку використаних джерел наведена використана при написанні проекту література.

Додаток А містить DDL-скрипт для бази даних інформаційної системи.

Пояснювальна записка складена згідно Стандарту підприємства.

Дана пояснювальна записка містить 110 с., 30 малюнків, 7 таблиць, 1 додаток.

Ключові слова: ІНФОРМАЦІЯ, ФОРМАТ, ПРОЕКТ, ТАБЛИЦЯ, ДІАГРАМА, БАЗА ДАНИХ, ІНФОРМАЦІЙНА СИСТЕМА, DDL-СКРИПТ, МОДЕЛЬ, АВТОМАТИЗОВАНА СИСТЕМА, МЕТОДОЛОГІЯ, ТАБЛИЦЯ, С++, UML-ДІАГРАМА, БАЗА ДАНИХ, ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ, СИСТЕМА КЕРУВАННЯ БАЗАМИ ДАНИХ, SQL-ЗАПИТ, ПРОГРАМНИЙ ПРОДУКТ, АВТОМАТИЗОВАНА СИСТЕМА.

Зміст

[Вступ 8](#_Toc450119186)

[1 ЗАГАЛЬНИЙ РОЗДІЛ 10](#_Toc450119187)

[1.1 Вивчення об’єкту дослідження 10](#_Toc450119188)

[1.1.1 Коротка інформація про підприємство 10](#_Toc450119189)

[1.1.2 Підрозділи і користувачі системи 12](#_Toc450119190)

[1.1.3 Бачення виконання проекту і границі проекту 12](#_Toc450119191)

[1.2 Аналіз існуючого рівня автоматизації 13](#_Toc450119192)

[1.3 Загальні вимоги до АІС 14](#_Toc450119193)

[1.3.1 Функції та бізнес-процеси 14](#_Toc450119194)

[1.3.2 Матриця організаційної відповідальності 19](#_Toc450119195)

[1.3.3 Обладнання та інструменти 20](#_Toc450119196)

[1.3.4 Матриця відображення обладнання та інструментів на функції та бізнес-процеси 20](#_Toc450119197)

[1.3.5 Регламентуючі документи 21](#_Toc450119198)

[1.3.6 Матриця відображення регламентуючих документів на функції та бізнес-процеси 22](#_Toc450119199)

[1.4 Огляд і аналіз існуючих методів і засобів вирішення задач 24](#_Toc450119200)

[1.5 Постановка задачі 28](#_Toc450119201)

[2. СПЕЦІАЛЬНИЙ РОЗДІЛ 32](#_Toc450119202)

[2.1 Проектування підсистеми, що розробляється 32](#_Toc450119203)

[2.1.1 Методи та засоби структурного системного аналізу та проектування 32](#_Toc450119204)

[2.1.2 Методи та засоби об'єктно-орієнтованого аналізу і проектування 48](#_Toc450119205)

[2.2 Інформаційне забезпечення 56](#_Toc450119206)

[2.2.1 Концептуальна модель АІС 58](#_Toc450119207)

[2.2.2 Даталогічна модель АІС 60](#_Toc450119208)

[2.3 Математичне забезпечення 60](#_Toc450119209)

[2.3.1 Математичні моделі системи 60](#_Toc450119210)

[2.3.2 Розробка алгоритмів рішення функціональної задачі 60](#_Toc450119211)

[2.3.3 Визначення та оцінка якісних показників алгоритмів, порівняння з існуючими 70](#_Toc450119212)

[2.4 Програмне забезпечення 70](#_Toc450119213)

[2.4.1 Схема взаємодії програмних модулів 74](#_Toc450119214)

[2.4.2 Архітектура системи 74](#_Toc450119215)

[2.5 Організаційне забезпечення 76](#_Toc450119216)

[2.5.1 Інструкція користувача 76](#_Toc450119217)

[2.5.2 Інструкція адміністратора 78](#_Toc450119218)

[2.5.3 Інші інструкції (фахівця, програміста, системного програміста) 79](#_Toc450119219)

[2.6 Технічне забезпечення 79](#_Toc450119220)

[2.6.1 Обґрунтування вибору (розробки) технічного забезпечення АІС 79](#_Toc450119221)

[2.6.2 Структура мережної системи (топологічна і логічна) 80](#_Toc450119222)

[2.6.3 Структура комплексу засобів автоматизації 81](#_Toc450119223)

[2.6.4 Специфікація обладнання 81](#_Toc450119224)

[3 ОХОРОНА ПРАЦІ 82](#_Toc450119225)

[Висновок 83](#_Toc450119226)

[Перелік використаних джерел 85](#_Toc450119227)

[Додаток А DDL-скрипт 86](#_Toc450119228)

**Перелік основних позначень і скорочень**

IDEF0 - методологія функціонального моделювання

SQL - структурована мова запитів

АІС - автоматизована інформаційна система

ЕОМ - електронно-обчислювальна машина

ПІБ - прізвище ім`я по батькові

ПК - персональний комп'ютер

BPwin - CASE-засіб

IDEF3 - функціональні моделі технологічних процесів

DFD - функціональні моделі інформаційних систем

FEO - діаграми тільки для експозиції

EPC - подієвий ланцюжок процесів

BPMN - методологія моделювання

# Вступ

Обчислювальну техніку в останні роки широко використовують в усіх сферах діяльності людей. Вона стала каталізатором науково-технічного прогресу. Необхідною є потреба в оволодінні знаннями і навичками використання цієї техніки. Кожній цивілізації доводиться мати справу з обробкою інформації. З розвитком економіки і зростанням чисельності населення зростає і обсяг взаємозв'язаних даних, необхідних для вирішення комерційних і адміністративних завдань. Взаємопов'язані дані називають інформаційною системою. Така система в першу чергу покликана полегшити працю людини, але для цього вона повинна якомога краще відповідати дуже складної моделі реального світу.

Ядром інформаційної системи є збережені в ній дані. На будь-якому підприємстві дані різних відділів, як правило, перетинаються, тобто використовуються в декількох підрозділах або взагалі є загальними. Наприклад, для мети управління часто потрібна інформація по всьому підприємству. Що зберігаються в інформаційній системі дані повинні бути легко доступні в такому вигляді, в якому вони потрібні для конкретної виробничої діяльності підприємства. При цьому не має істотного значення спосіб зберігання даних. Сьогодні на підприємстві ми можемо зустріти систему обробки даних традиційного типу, в якій службовець вручну поміщає дані в швидкозшивач, і поряд з нею - сучасну систему із застосуванням самої швидкодіючої ЕОМ, найскладнішого обладнання і програмного забезпечення. Незважаючи на вражаючу несхожість, обидві ці системи зобов'язані надавати достовірну інформацію у визначений час, певній особі, в певному місці і з обмеженими витратами.

Інформаційна система - великі масиви даних про об'єкти і явища реального світу разом з програмно-апаратними засобами їх обробки. За останні двадцять років значно зріс обсяг і обіг інформації в усіх сферах життєдіяльності людини: економічної, фінансової, політичної, духовної. І процес накопичення, обробки та використання знань постійно прискорюється. Вчені стверджують, що кожні десять років кількість інформації збільшується вдвічі. У зв'язку з цим виникає необхідність використання автоматичних засобів, що дозволяють ефективно зберігати, обробляти і розподіляти накопичені дані.

 Виходячи із сучасних вимог, що пред'являються до якості роботи фінансового ланки великого підприємства, не можна не відзначити, що ефективна робота його цілком залежить від рівня оснащення компанії інформаційними засобами на базі комп'ютерних систем автоматизованого обліку роботи різних підрозділів підприємства.

Комп'ютерний облік має свої особливості і радикально відрізняється від звичайного. Комп'ютер не тільки полегшує облік, скорочуючи час, потрібний на оформлення документів та узагальнення накопичених даних для аналізу ходу торгової діяльності, необхідного для управління нею. Звіти про становище в торгівлі, одержувані за допомогою комп'ютера, можна отримати і без нього - ніякої особливої ​​математики в комп'ютері не міститься - але на розрахунки піде стільки часу, що вони вже ні на що не будуть потрібні; або ними доведеться зайняти таку кількість розраховувачів, що на їх зарплату піде значно більше, ніж буде отримано прибутку в результаті їх розрахунків. Таким чином при застосуванні комп'ютера "кількість переходить в якість»: збільшення швидкості розрахунків уможливлює якісне поліпшення самої схеми побудови торгівлі.

Метою даного дипломного проекту є розробка автоматизованої інформаційної системи обліку роботи стоматологічного, яка дозволить вести: облік роботи, матеріалів та інструментів стоматологічного центру. Вона допоможе адміністратору центру в обробці, пошуку, зберіганні інформації про пацієнтів центру, розкладі прийомів і роботи лікарів. Допоможе фахівцю - стоматологу у веденні медичної карти пацієнта, а так само медсестрі у веденні обліку матеріалів, інструментарію, та також інформації про лікування хвороби пацієнта.

# 1 ЗАГАЛЬНИЙ РОЗДІЛ

## 1.1 Вивчення об’єкту дослідження

### 1.1.1 Коротка інформація про підприємство

Стоматологія - розділ медицини, що займається вивченням зубів, їх будови і функціонування, їх захворювань, методів їх профілактики та лікування, а також хвороб порожнини рота, щелеп.

Метою діяльності стоматологічного центру «АВАС - ДЕНТ» є виконання лікувальних та профілактичних заходів сучасної стоматології :

* комплексна діагностика і повне обстеження порожнини рота з складанням плану лікування;
* професійна гігієна порожнини рота;
* професійне відбілювання зубів;
* терапевтичне лікування зубів;
* сучасна ендодонтія (лікування коренів зубів);
* сучасна парадонтологія (лікування захворювань ясен – пародонтиту, гінгівіту, пародонтозу);
* естетична реставрація зубів, мікропротезування;
* ортопедія (протезування);
* хірургія;
* ортодонтичне лікування дорослих і дітей (виправлення прикусу).

Основне завдання даного центру є задоволення пацієнтів умовами надання послуг: діагностики, лікування, протезування, операції, а також контроль післяопераційного періоду пацієнтів.

Розглянута приватний центр має у своєму складі керуючого, який виконує обов'язки менеджера з персоналу і обов'язки головного лікаря. Також в клініці працює лікар - стоматолог, що спеціалізується на протезуванні зубів та операціях, медсестри, які є асистентами на операціях, роблять рентгенівські знімки та здійснюють їх обробку, також в їх обов'язки входить догляд за пацієнтами в період їх реабілітації в післяопераційні години, а так само прибирання приміщень, вони відповідають за своєчасну поставку медикаментів, медичних інструментів та обладнання. Адміністратор центру реєструє клієнтів і оформляє медичні карти. Бухгалтер веде бухгалтерські записи, розраховує вартість послуг і є одночасно касиром. Вся документація ведеться в паперовому вигляді.

Більш конкретно періодично повторюваний процес надання послуг в галузі стоматології зводиться до наступного. Клієнт приходить в центр або телефонує і дізнається всю, потрібну йому інформацію. В результаті отриманих відомостей клієнт приймає рішення стати пацієнтом центру, або відмовитися від пропонованих послуг. Якщо клієнт стає пацієнтом центру, він записується на відвідування стоматологічного кабінету. Потім в призначений час йде на прийом до стоматолога. Лікар-стоматолог проводить огляд. За результатами огляду лікар виносить висновок про способи подальшого лікування і методів, якими воно буде здійснюватися. У разі необхідності хірургічного втручання лікар направляє пацієнта на здачу аналізів. Якщо передбачуване лікування має на увазі операцію, то лікар-стоматолог пояснює пацієнту процес операції та можливі ускладнення після операції і передбачувану вартість. Якщо пацієнта влаштовують умови надання послуги, на пацієнта заводиться в реєстратурі особиста картка, підписується договір, що передбачає різні нюанси, які можуть виникнути в процесі проведення операції та можливі післяопераційні ускладнення. Потім оплачується половина вартості послуги. І пацієнт направляється: діагностика, операція і післяопераційний спостереження. У разі якщо після операції, пацієнта влаштовує результат зробленої операції, оплачується друга половина за операцію. Центр має рентгенівський кабінет.

Персонал стоматологічного центру складається з головного лікаря, лікарів стоматологів першої та другої категорії, медсестер, адміністраторів. Професія стоматолога включають в себе терапевта, хірурга, стоматолога - ортопеда і ортодонта.

Терапевт - проводить первинний огляд, ставить діагноз, лікує зуби, направляє на рентген, готує порожнину рота під протезування .

Хірург - в його обов'язки входять щелепні операції, виправлення щелепних дефектів, видалення зубів.

Ортопеди стоматологи, які займаються протезуванням: встановлюють коронки, мости.

Ортодонт займається проблемами з прикусом.

Обов'язковими умовами стоматологічної центру є: наявність ліцензій на надавані послуги; наявність різних дипломів та сертифікатів лікарів; кваліфікація стоматологів, послуги стоматологів вузької спеціалізації; чистота і гігієна приміщення, використання одноразових і стерильних приладів, надання бахіл, у лікарів - марлеві пов'язки; прийом пацієнта проводиться в «чотири руки», наявність асистентів; використовується сучасне обладнання; для лікування вагітних жінок використовуються тільки спеціальні препарати і крісла; головним гаслом є «Видалення тільки в крайньому випадку!»; надання гарантій на всі види послуг.

### 1.1.2 Підрозділи і користувачі системи

Структура стоматологічної поліклініки

* + Головний лікар:
  + Відділ кадрів:
    - Начальник відділу кадрів
  + Лікувальний відділ
    - Заступник головного лікаря з медичної роботи
      * Стоматолог-терапевт
      * Медсестра
      * Адміністратор
  + Бухгалтерія
    - Головний бухгалтер
      * Бухгалтер
  + Юридична служба
    - Начальник юридичної служби
  + Господарська частина
    - Заступник головного лікаря з господарської роботи

### 1.1.3 Бачення виконання проекту і границі проекту

Використовуючи розроблену інформаційну систему адміністратор зможе:

* при первинному відвідуванні центру пацієнтом, заводити медична карту, куди вносяться усі контактні дані пацієнта;
* надавати попередню інформацію про то, які надають послуги, їх вартість, порядок оплати безпосередньо або по телефону, використовуючи електронний прейскурант;
* вносити дані про прийом пацієнтів на консультацію, первинний огляд та лікування;
* скласти графік прийому, ґрунтуючись на графіку роботи лікарів та за згодою із пацієнтом;
* взаємодіяти з пацієнтом при оплаті лікувальних або профілактичних заходів, включаючи безготівкову оплату, використовуючи систему знижок для постійних клієнтів, видавати гарантії та чеки в друкованому вигляді, за допомогою журналу оплати і розрахунку собівартості послуг, прийомів, лікування.
* вести реєстр наданих послуг за прейскурантом.

Використовуючи розроблену інформаційну систему стоматолог зможе:

* всю інформацію, отриману при огляді пацієнта, вводити у форму медичної карти;
* реєструвати, вводити причини та умови протікання хвороб, алергічних реакцій пацієнта на застосовувані препарати;
* планувати лікування (терміни, вартість, показання до вибору способів лікування), використовуючи електронний прейскурант.

Використовуючи розроблену інформаційну систему медсестра зможе:

* реєструвати всі результати обстежень та виконаних медичних послуг в електронній медичній документації;
* фіксувати діагноз і проведене лікування в медичній карті амбулаторного хворого;
* вести контроль витрат матеріалів в електронній формі;
* вести автоматизований облік запасів матеріалів і інструментарію, які надійшли та знаходяться на складі;
* вести журнал оновлень і списань матеріалів та інструментів.

Система автоматизує роботу адміністратора, стоматолога, медсестрі. За рамки проекту вийдуть загальні процеси стоматологічного центру. Система – для трьох користувачів: стоматолога, адміністратора центру, медсестрі.

## 1.2 Аналіз існуючого рівня автоматизації

Список програмного забезпечення, яке використовується компанією на момент обстеження:

* Власна розробка для загального керування центром;
* Excel для планування закупок.

Існуючий рівень автоматизації в таб. 1.2

Таблиця 1.2 – Існуючий рівень автоматизації:

|  |  |
| --- | --- |
| Об'єкти автоматизації | Наявність |
| Кількість робочих станцій, всього: | 1 |
| Кількість співробітників відділу ІT | 1 |
| Кількість ПК, що одночасно працюють у мережі | 1 |
| Наявність і форма зв'язку з віддаленими об'єктами | - |
| Кількість робочих станцій на віддаленому об'єкті | - |
| Характеристики комп'ютерів | Від Celeron 600 і вище |
| Операційна система | Windows 7, 8 |
| Системи, які представляється можливим залишити без зміни | власна розробка для загального керування центром |

## 1.3 Загальні вимоги до АІС

Ключові функціональні вимоги до інформаційної системи:

* засоби захисту даних від несанкціонованого доступу.
* розмежування доступу до даних відповідно до посадових обов'язків;
* керування запасами. Оперативне одержання інформації про залишки на складі;
* керування наданням стоматологічних послуг пацієнтам
* керування закупками;
* планування закупок у розрізі постачальників;
* контроль взаєморозрахунків з постачальниками і клієнтами;
* одержання звітів.

### 1.3.1 Функції та бізнес-процеси

Бізнес-процеси веденняосновної діяльності

1.Матеріально-технічне забезпечення діяльності стоматологічного центру:

* + Планування і придбання необхідних ресурсів
    - Вибір і сертифікація постачальників
    - Купівля медичного обладнання
    - Купівля сировини і матеріалів
    - Придбання технологій
  + Управління процесом закупки
    - Укладання контрактів
    - Оформлення замовлень
    - Моніторинг поставок
  + Транспортна доставка
  + Організація зберігання
    - Забезпечення роботи складів
    - Контроль за станом складів
    - Забезпечення збереження
  + Виробничі процеси
  + Надання послуг пацієнту
    - спеціальні вимоги до роботи стоматологічного центру
    - ведення картотеки пацієнтів;
    - планування прийому пацієнтів;
    - планування роботи лікарів;
    - взаєморозрахунки з пацієнтами;
    - ведення штатного розкладу клініки, кадровий облік;
    - ведення карт стану зубів;
    - аналізи пацієнта;
    - ведення історії хвороби пацієнта ;
    - планування профілактичних оглядів;
    - ідентифікація та планування надання стоматологічних послуг пацієнтам
    - забезпечення усього необхідного для обслуговування пацієнта
    - надання послуг та засобів лікування пацієнту
    - надання послуг та засобів профілактики
  + Взаємодія в стоматологічному центрі
    - ведення прейскуранта послуг клініки
    - розробка і ведення планів лікування пацієнтів
    - облік витрати стоматологічних матеріалів та інструментів ;
    - планування закупок матеріалів;
    - друк статистичних звітів ;
    - роздруківка типових бланків документів , згідно регіональних і медичних стандартів ;
    - організація взаємодії між лікарями і регістратурой
  + Забезпечення гарантії якості наданих послуг пацієнту
  + Формування необхідних трудових ресурсів (фахівців)
    - Формування необхідних кваліфікаційних вимог
    - Виявлення необхідності в навчанні та проведення навчання
    - Відстеження і управління підвищенням кваліфікації

2. Маркетинг і продажі

* + Визначення (виявлення) потреб і потреб (очікувань) споживачів
  + Вимірювання задоволеності споживачів послугами, сервісним обслуговуванням
  + Відстеження змін на ринку або змін в очікуваннях споживачів
  + Збут послуг
  + Стратегія ціноутворення
    - Рекламна стратегія
    - Виявлення цільових споживачів і їх потреб
    - Розробка прогнозу продажів
    - Продаж послуг
    - Управління портфелем продукції / послуг
  + Відповіді на запити пацієнтів

Бізнес-процесидопоміжні

1. Підтримка інфраструктури фірми (загальне управління, планування, фінансування, бухгалтерський облік, юридичне забезпечення.)
   * Управління фінансовими ресурсами
     + Розробка бюджетів
     + Управління розміщенням ресурсів
     + Планування ризикового капіталу
     + Управління грошовими потоками
     + Управління структурою капіталу
     + Управління фінансовим ризиком
   * Обробка фінансових і бухгалтерських операцій
   * Формування звітної інформації
     + Підготовка зовнішньої фінансової інформації
     + Підготовка внутрішньої фінансової інформації
   * Проведення внутрішнього аудиту
   * Управління податками
     + Планування податкової стратегії
     + Використання технологій і прийомів податкового менеджменту
     + Управління розбіжностями з питань оподаткування
     + Робота з податковими відомствами
   * Управління матеріальними ресурсами
     + Управління плануванням капіталу
     + Придбання та перегрупування основних засобів
     + Управління ресурсами
   * Управління зовнішніми зв'язками
2. Інженерно-технічне забезпечення (утримання офісів, будівель, виробничих корпусів .)
3. Інформаційне забезпечення
   * Планування управління інформаційними ресурсами
     + Формулювання вимог на основі стратегії центру
     + Визначення архітектури
     + Планування та прогнозування інформаційних технологій
     + Встановлення загальних для підприємства стандартів даних
     + Встановлення стандартів якості та контролю
   * Розробка і розгортання допоміжних систем підприємства
     + Проведення аналізу специфічних потреб
     + Вибір інформаційних технологій
     + Розробка
     + Тестування, оцінка та впровадження
   * Впровадження систем контролю та безпеки
   * Управління зберіганням і отриманням інформації
   * Управління інформаційними ресурсами та операціями
   * Оцінка і контроль якості інформації
4. Документообіг
5. Управління персоналом:
   * Створення і управління стратегією людських ресурсів
   * Аналіз і планування рівня виробництва
   * Розвиток та підготовка працівників
   * Розробка інформаційних систем управління персоналом
6. Економічна безпека
7. Екологія:

* Розробка стратегії охорони навколишнього середовища
* Забезпечення відповідності нормам
* Розробка та виконання програм запобігання забруднення і захисту навколишнього середовища
* Розробка та впровадження інформаційної системи моніторингу та контролю
* Управління відносинами з державними органами та громадськістю з екологічних питань

Бізнес-процесирозвитку і вдосконалення

1. Стратегічне управління
   * Відстеження стану зовнішнього середовища
     + Аналіз та вивчення конкуренції
     + Виявлення тенденцій в економіці (галузі)
     + Виявлення політичних і регулюючих подій
     + Оцінка технологічних інновацій
     + Виявлення соціальних і культурних змін
   * Визначення концепції бізнесу і стратегії організації
   * Розробка структури організації і відносин між організаційними одиницями
   * Розробка і встановлення цілей організації
   * Бізнес-планування
2. Розвиток технологій (вдосконалення процесу послуг, НДДКР, проектування, дизайн)
   * Розробка концепції та планів випуску нового послуги
   * НДДКР (розробка, створення і оцінка дослідних зразків)
   * Удосконалення існуючих послуг
   * Підготовка до виробництва (впровадженню)
3. Управління проектами
   * Управління процесом розробки
   * Управління проектами реорганізації
   * Управління маркетинговими проектами
4. Управління якістю
   * Оцінка показників діяльності організації
     + Створення системи оцінки
     + Оцінка якості послуг
     + Оцінка витрат на забезпечення якості
   * Проведення оцінки якості
   * Порівняння з конкурентами та ін. компаніями (бенчмаркінг)
   * Поліпшення процесів і систем
     + Впровадження постійного процесу поліпшень
     + Моніторинг процесів і систем
     + Реінжиніринг процесів
   * Впровадження тотального управління якістю

#### Опис бізнес процесів, що будуть автоматизовані:

1. Надання послуг пацієнту

2. Дуже важливий

3. Прийом пацієнтів

4. Перелік функцій бізнес-процесу:

* спеціальні вимоги до роботи стоматологічного центру
* ведення картотеки пацієнтів;
* планування прийому пацієнтів;
* планування роботи лікарів;
* взаєморозрахунки з пацієнтами;
* ведення штатного розкладу клініки, кадровий облік;
* ведення карт стану зубів;
* аналізи пацієнта;
* ведення історії хвороби пацієнта ;
* планування профілактичних оглядів;
* ідентифікація та планування надання стоматологічних послуг пацієнтам;
* забезпечення усього необхідного для обслуговування пацієнта;
* надання послуг та засобів лікування пацієнту;
* надання послуг та засобів профілактики.

5. Прийом пацієнта, його дані, необхідні матеріали та інструменти.

Цю функцію виконує стоматолог, адміністратор, медсестра, взаємодіючі між собою, за допомогою використання медичного обладнання та канцелярії, щоб надати пацієнту необхідну допомогу в лікуванні та профілактиці зубної порожнини.

6. Ефективне лікування чи профілактика.

7. В залежності від складності роботи та стану ротової порожнини, зубів можливі певні проблеми, які впливають на час, вартість надання послуги.

8. Використання іншого плану лікування та профілактики.

9.1 раз на годину.

10.Більш менш 20 хв.

Наступний:

1. Взаємодія в стоматологічному центрі

2. Дуже важливий

3. Перед прийомом пацієнтів

4. Перелік функцій бізнес-процесу:

* ведення прейскуранта послуг клініки
* розробка і ведення планів лікування пацієнтів
* облік витрати стоматологічних матеріалів та інструментів ;
* планування закупок матеріалів;
* друк статистичних звітів ;
* роздруківка типових бланків документів , згідно регіональних і

медичних стандартів ;

* організація взаємодії між лікарями і регістратурою.

5. Прийом пацієнта, його дані, необхідні матеріали та інструменти.

Цю функцію виконує стоматолог, адміністратор, медсестра, взаємодіючі між собою, за допомогою використання медичного обладнання та канцелярії, щоб надати пацієнту необхідну допомогу в лікуванні та профілактиці зубної порожнини.

6. Ефективне лікування чи профілактика.

7. В залежності від складності роботи та стану ротової порожнини, зубів можливі певні проблеми, які впливають на час, вартість надання послуги.

8. Використання іншого плану лікування та профілактики.

9.1 раз на годину.

10.Більш менш 20 хв.

### 1.3.2 Матриця організаційної відповідальності

Виробничі процеси на таб. 1.3.2

Таблиця 1.3.2 - Матриця організаційної відповідальності

|  |  |  |  |
| --- | --- | --- | --- |
| Список функцій та бізнес-процесів | Адміністратор | Стоматолог | Медсестра |
| спеціальні вимоги до роботи стоматологічного центру | + |  |  |
| ведення картотеки пацієнтів | + |  | + |
| планування прийому пацієнтів | + |  |  |
| планування роботи лікарів | + |  |  |
| взаєморозрахунки з пацієнтами | + |  |  |
| ведення штатного розкладу клініки, кадровий облік | + | + |  |
| ведення карт стану зубів |  | + | + |
| аналізи пацієнта |  |  | + |
| ведення історії хвороби пацієнта |  | + | + |
| планування профілактичних оглядів; |  | + |  |
| ідентифікація та планування надання стоматологічних послуг пацієнтам |  | + |  |
| забезпечення усього необхідного для обслуговування пацієнта |  |  | + |
| надання послуг та засобів лікування пацієнту |  | + |  |
| надання послуг та засобів профілактики |  | + |  |
| ведення прейскуранта послуг клініки | + |  |  |
| розробка і ведення планів лікування пацієнтів |  | + |  |
| облік витрати стоматологічних матеріалів та інструментів |  |  | + |
| планування закупок матеріалів |  |  | + |
| друк статистичних звітів | + |  | + |
| роздруківка типових бланків документів | + |  | + |
| організація взаємодії між лікарями і регістратурою |  |  | + |

### 1.3.3 Обладнання та інструменти

* медичне обладнання
* канцелярські інструменти
* ПК і периферійний пристрій

### 1.3.4 Матриця відображення обладнання та інструментів на функції та бізнес-процеси

Матриця відображення обладнання та інструментів на функції представлена на таблиці 1.3.4

Таблиця 1.3.4 - Матриця відображення обладнання та інструментів на функції

|  |  |  |  |
| --- | --- | --- | --- |
| Список функцій та бізнес-процесів | медичне обладнання | канцелярські інструменти | ПК і периферійний пристрій |
| спеціальні вимоги до роботи стоматологічного центру |  | + | + |
| ведення картотеки пацієнтів |  | + |  |
| планування прийому пацієнтів |  | + |  |
| планування роботи лікарів |  | + |  |
| взаєморозрахунки з пацієнтами |  | + | + |
| ведення штатного розкладу клініки, кадровий облік |  | + |  |
| ведення карт стану зубів |  | + |  |
| аналізи пацієнта | + | + |  |
| ведення історії хвороби пацієнта |  | + |  |
| планування профілактичних оглядів; |  | + |  |
| ідентифікація та планування надання стоматологічних послуг пацієнтам |  | + |  |
| забезпечення усього необхідного для обслуговування пацієнта | + | + | + |
| надання послуг та засобів лікування пацієнту | + |  |  |
| надання послуг та засобів профілактики | + |  |  |

Продовження Таблиця 1.3.4

|  |  |  |  |
| --- | --- | --- | --- |
| ведення прейскуранта послуг клініки |  |  | + |
| розробка і ведення планів лікування пацієнтів |  | + | + |
| облік витрати стоматологічних матеріалів та інструментів |  | + | + |
| планування закупок матеріалів |  | + | + |
| друк статистичних звітів |  | + | + |
| роздруківка типових бланків документів |  | + | + |
| організація взаємодії між лікарями і регістратурою |  | + | + |

### 1.3.5 Регламентуючі документи

вхідні:

- журнал прийому пацієнтів;

- процеси та процедури взаємодії в стоматологічному центрі

- накладні;

вихідні:

- звіти,

- закази постачальникам;

- чек;

- маркетингові матеріали;

Журнал прийому пацієнтів – документ, для запису пацієнтів на прийом до стоматолога.

Накладні – документ з інформацією про заказані матеріали та інструменти.

Звіти - усі звіти про роботу стоматологічного центру.

Закази постачальникам – оформлений список необхідних матеріалів та інструментів на закупку до постачальників.

Чек- надані послуги пацієнтам.

Маркетингові матеріали - презентації, виставки, реклама, маркетингові дослідження.

Правила і процедури - різні правила і процедури, якими керується стоматологічний центр в процесі своєї діяльності (правила поведінки з медичним обладнанням, процедура спілкування з клієнтами).

### 1.3.6 Матриця відображення регламентуючих документів на функції та бізнес-процеси

Матриця відображення регламентуючих документів на функції представлена на таблиці 1.3.6.

Таблиця 1.3.6 - Матриця відображення регламентуючих документів на функції

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Список функцій та бізнес-процесів | журнал прийому пацієнтів | | процеси та процедури | | накладні | | чек | | звіт | | закази | | матеріали | |
| спеціальні вимоги до роботи стоматологічного центру |  | | + | | + | |  | | + | |  | | + | |
| ведення картотеки пацієнтів | + | | + | |  | |  | | + | |  | | + | |
| планування прийому пацієнтів | | + | | + | |  | |  | | + | |  | |  |
| планування роботи лікарів | |  | | + | |  | |  | | + | |  | | + |
| взаєморозрахунки з пацієнтами | | + | | + | |  | | + | | + | |  | | + |
| ведення штатного розкладу клініки, кадровий облік | |  | | + | |  | |  | | + | |  | | + |
| ведення карт стану зубів | | + | | + | |  | |  | | + | |  | |  |
| аналізи пацієнта | | + | | + | |  | |  | | + | |  | |  |
| ведення історії хвороби пацієнта | | + | | + | |  | |  | | + | |  | |  |
| планування профілактичних оглядів; | | + | | + | |  | |  | | + | |  | | + |
| ідентифікація та планування надання стоматологічних послуг пацієнтам | | + | | + | |  | |  | | + | |  | | + |
| забезпечення усього необхідного для обслуговування пацієнта | |  | | + | | + | |  | | + | | + | |  |

Продовження Таблиці 1.3.6

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| надання послуг та засобів лікування пацієнту | + | + | + | + | + |  | + |
| надання послуг та засобів профілактики | + | + | + | + | + |  | + |
| ведення прейскуранта послуг клініки |  | + | + |  | + | + | + |
| розробка і ведення планів лікування | + | + | + | + | + | + | + |
| облік витрати матеріалів та інструментів | + | + | + |  | + | + | + |
| планування закупок матеріалів |  | + | + |  | + | + | + |
| друк статистичних звітів |  | + | + | + | + | + | + |
| роздруківка бланків |  | + | + | + | + | + | + |
| організація взаємодії між лікарями |  | + | + |  | + |  | + |

## 1.4 Огляд і аналіз існуючих методів і засобів вирішення задач

Розглянемо готові рішення, здатні замінити розроблювальне програмне забезпечення за темою «АІС обліку роботи стоматологічного центру « АВАС - ДЕНТ » ». Серед них можна відзначити

* + DentExpert;
  + SharkDental;
  + Дентал-Софт.

Програмний продукт DentExpert

Система DentExpert - спеціалізована автоматизована система керування стоматологічною клінікою, яка призначена для впровадження в стоматологічних клініках будь-яких масштабів.

Система керування стоматологічною клінікою DentExpert (система комплексної автоматизації стоматологічної клініки) призначена для автоматизації процесів планування, обліку та аналізу роботи приватних стоматологічних клінік, а також впровадження сучасних інформаційних і цифрових технологій в практику специфічних медичних процесів.

Невеликі окремі стоматологічні програми дозволять вирішити окремі питання в клініці, але не дозволяють організувати роботу колективу, структурувати і централізувати інформацію, забезпечити доступ до неї всіх зацікавлених осіб, контролювати співробітників і важливі моменти стоматологічного процесу і в кінцевому підсумку забезпечити керівника достатньою інформацією для прийняття рішення.

Система DentExpert забезпечує автоматизацію функціонування та взаємодії наступних напрямків діяльності клініки:

* управління клінікою;
* робота зі стоматологічними послугами;
* робота реєстратури;
* робота бухгалтерії;
* робота складу.

У ці напрямки входять такі групи завдань:

* реєстратура та запис на прийом;
* планування і виконання лікування;
* обстеження та лабораторні аналізи;
* облік оплати та розрахунки з пацієнтами;
* склад, облік і списання матеріалів;
* розрахунок зарплати;
* рентгенівські знімки;
* робота з організаціями та страховими компаніями;
* облік реклами та оцінка її ефективності;
* друк документів з налаштованим шаблонами;
* загальний статистичний облік і аналіз;
* статистичний аналіз ефективності роботи;
* адміністрування системи, управління доступом і аудит;
* резервне копіювання і відновлення даних.

Використання системи DentExpert, за рахунок автоматизації клініки, дозволяє:

* підвищити ефективність взаємодії різних підрозділів і посадових осіб стоматологічної клініки;
* уніфікувати і систематизувати дані про пацієнтів, персонал і устаткування клініки, проведених роботах, використаних матеріалах, розрахунках з клієнтами;
* планувати роботу персоналу, прийом пацієнтів, закупівлю стоматологічних та витратних матеріалів.

В системі DentExpert реалізовано кілька рівнів безпеки даних, що стосуються розрахункових операцій з клієнтами, обліку витрачених матеріалів та конфіденційної медичної інформації про пацієнтів стоматологічної клініки.

Забезпечена можливість обміну даними (імпорт/експорт) при операціях системи DentExpert з різними системами автоматизації бухгалтерського обліку.

Забезпечено сполучення системи DentExpert з сучасними зразками стоматологічного діагностичного обладнання для отримання, зберігання і обробки графічної і відеоінформації.

Система DentExpert складається з декількох функціональних підсистем. Використання можливостей кожної з підсистем визначається на рівні адміністрування системи персонально для кожного користувача.

Програмний продукт SharkDental

SharkDental - автоматизований облік в стоматологічному кабінеті, клініці. Стоматологічна програма для приватних кабінетів і невеликих клінік. Містить реєстраційну картку пацієнта, зубну формулу, модуль роботи з рентгенівськими і цифровими знімками, редагований список маніпуляцій, графік роботи лікарів, розбивку по кабінетах, можливість роботи по мережі.

Основні функції програми:

* автоматизувати роботу реєстратури;
* зберігати в електронному вигляді повну картотеку пацієнтів , що містить стоматологічні картки хворих і приєднані до них бібліотеки графічних зображень (рентгенологічних або цифрових знімків ) зубів і ротової порожнини пацієнтів;
* формувати звіти для проведення оперативного аналізу діяльності установи за будь-який обраний проміжок часу;
* роздруковувати необхідну інформацію.

Фактично весь функціонал програми зосереджений всього лише в двох вікнах - головному вікні, і в "Стоматологічної картці".

Програма складається з «головного» вікна, що включає кілька закладок, основного меню і двох панелей інструментів, на яких для зручності про дубльовані пункти меню. Деякі пункти меню також про дубльовані у спливаючому / контекстному меню, яке викликається при натисканні правої кнопки миші у списку.

Одна з панелей інструментів (перша після меню) є спільною для всіх закладок, тобто при натисканні на кнопки панелі виконувані дії залежать від обраної закладки.

Друга панель призначена для виконання операцій з графіком прийому.

Головне вікно містить в собі меню, панель з кнопками маніпуляції поточним довідником, панель з кнопками запису на прийом, панель фільтра по лікаря. Все інше місце займає набір закладок для навігації по довідниках програми: журнал прийому, журнал пацієнтів, користувачі, маніпуляції, лікарі.

Друге вікно, "Стоматологічна картка", яке викликається за подвійним клацанням миші на імені пацієнта, містить у собі дві закладки - загальні дані та історія хвороби.

Є можливість перемикатися між цими двома вікнами, однак зміна активної записи в "Журналі пацієнтів" не призводить до зміни відображуваної картки, більш того, є можливість запуску необмеженої кількості вікон " Стоматологічної картки".

Програма в більшості випадків управляється мишею, проте в деяких місцях акцент зроблений на введення з клавіатури. У деяких довідниках введення даних зроблено прямо в таблиці.

В якості переваги програми можна відзначити те, що інтерфейсні елементи запам'ятовують своє становище після виходу з програми; функцію "Пошук", яка дозволяє відфільтрувати деякі таблиці по обраному полю; можливість вказівки полів для відображення в таблицях; можливість експорту таблиць в Exсel .

Список картотеки можна сортувати по будь-якому полю.

Список " Журнал прийому " містить можливість переглянути, додати або видалити запис на прийом за конкретною датою, тобто після вибору "Дати прийому " програма відобразить список прийомів на цю дату.

Як додаткові функції можна розглядати можливість зробити вибірку за вказаною в "Фільтрі" лікарю, і реалізовану закладку "Пацієнти на сьогодні" для отримання вибірки пацієнтів на поточний день. Запис на прийом містить в собі П.І.Б. пацієнта (і автоматично прикріплює до нього лікаря), дату і час прийому, а також примітки.

У програмі передбачено додавання нової картки прямо з діалогу створення запису на прийом.

Програмний продукт Дентал-Софт

Програма призначена для автоматизації стоматології - автоматизації документообігу стоматологічного кабінету та ведення електронних медичних карт стоматологічних пацієнтів.

У програмі для стоматологічної клініки відсутня необхідність для встановлених компонентів Microsoft Office. Вже існує вбудована мова програмування і дизайнер для формування власних звітів, користувач може змінювати / створювати звіти під свою клініку. Мається статистична звітність, будь-який звіт можна вивантажити в MS- Excel.

Розрахунок суми оплати в касу виконується автоматично, можливий розрахунок авансу з відображенням записи в медичній карті і подальшим його заліком при розрахунку лікувальної роботи по факту. За бажанням можна вести графік роботи лікарів і попередній запис пацієнтів на прийом, а так само направлення на аналізи і подальший введення результатів цих аналізів.

Можна вести облік за дисконтними картками, з автоматичним підрахунком знижок, залежно від накопиченої суми на карті.

Стоматологічна програма веде облік відвідуваності пацієнта і сама автоматично підраховує коли пацієнт первинний і коли повторний, в разі первинного відвідування пропонується вказати зубну формулу на початок року і описати об'єктивні дані первинного пацієнта. Є можливість прив'язати знімки до пацієнта з візіографа. Після написання номера картки пацієнта , в щоденнику лікарського обстеження, відразу відіб'ються всі рентген знімки даного пацієнта.

Передбачена можливість відображення в медичній карті стоматологічного пацієнта виписки лікарняного листа, його продовження,, закриття, дозволи видачі головним лікарем, виписки дубліката, порушення режиму.

Мається розширений пошук пацієнтів за фрагментами ПІБ, статі, народженими між датами, місцем роботи, професіями, контингенту, адресі, телефону та багато інших.

Контроль доступу до даних по групах користувачів, на рівні кожної форми / довідника.

Опис зубної формули здійснюється для всіх 6 порожнин зуба в графічному режимі.

Програмне забезпечення для стоматологів Дентал - Софт розбито на АРМ.

За замовчуванням пропонуються наступні АРМ -и:

* лікар - терапевт (хірург, пародонтолог);
* лікар - ортопед;
* лікар - ортодонт;
* рентген - лаборант;
* лікар - анестезіолог;
* реєстратура;
* довідники;
* каса;
* завідувач виробництвом;
* зубний технік;
* склад;
* реєстри;
* головний лікар.

У програму вже включені всі стоматологічні діагнози (терапевтичні, хірургічні, пародонтологічні, ортопедичні). Через АРМ “Довідники” ви можете надалі додавати (змінювати) нові діагнози, прив'язувати до них скарги, анамнез, об'єктивні дані, план лікування, види лікування, методи лікування й ціни.

Оцінка доцільності розробки нового програмного продукту. Зведемо отримані результати в підсумкову таблицю для прийняття рішення про доцільність розробки нового програмного продукту (Таблиця 1.4).

Програмний продукт 1: DentExpert не цілком задовольняє поставленому завданню і коштує дорого, так як автоматизує багато функцій, які не будуть використовуватися у стоматологічному центрі.

Програмний продукт 2: SharkDental не цілком задовольняє поставленому завданню, але коштує не так дорого, як інші системи.

Таблиця 1.4 - Зведені дані маркетингового дослідження

|  |  |  |
| --- | --- | --- |
| Найменування ПЗ | Відсоток відповідності поставленому завданню | Вартість |
| Програмний продукт 1: DentExpert | 85 % | 22464.6 |
| Програмний продукт 2: SharkDental | 60 % | 7188.6 |
| Програмний продукт 3: Дентал-Софт | 90 % | 13124.00 |
| Розроблюємий програмний продукт (АІС обліку роботи стоматологічного центру « АВАС - ДЕНТ » » | 100 % |  |

Програмний продукт 3:Дентал-Софт не досить задовольняє поставленому завданню і коштує дорого, так як автоматизує багато функцій, які не будуть використовуватися у стоматологічному центрі.

В результаті проведеного дослідження можна зробити висновок, що розробка нового програмного забезпечення доцільна. Для повноти дослідження необхідно провести оцінку економічної ефективності.

## 1.5 Постановка задачі

Найменування програми.: «Автоматизована інформаційна система обліку роботи стоматологічного центру «АВАС - ДЕНТ ».

Призначення програми. Автоматизована інформаційна система обліку роботи стоматологічного центру «АВАС - ДЕНТ» призначена для обліку роботи, матеріалів та інструментів стоматологічного центру. Вона допоможе адміністратору центру в обробці, пошуку, зберіганні інформації про пацієнтів центру, розкладі прийомів і роботи лікарів. Допоможе фахівцю - стоматологу у веденні медичної карти пацієнта, а так само медсестрі у веденні обліку матеріалів, інструментарію, та також інформації про лікування хвороби пацієнта.

Область застосування програми. Створювана інформаційна система може застосовуватися адміністраторами стоматологічних центрів, стоматологами і медсестрами з метою покращення та автоматизації їх повсякденних обов'язків.

Вимоги до функціональних характеристик програми: взаємодія користувачів з прикладним програмним забезпеченням , що входять до складу системи має здійснюватися за допомогою візуального графічного інтерфейсу (GUI). Інтерфейс системи повинен бути зрозумілим і зручним, не повинен бути перевантажений графічними елементами і повинен забезпечувати швидке відображення екранних форм. Навігаційні елементи повинні бути виконані в зручній для користувача формі. Засоби редагування інформації повинні задовольняти прийнятим угодам в частині використання функціональних клавіш, режимів роботи, пошуку, використання віконної системи. Введення-виведення даних системи, прийом керуючих команд і відображення результатів їх виконання повинні виконуватися в інтерактивному режимі. Інтерфейс повинен відповідати сучасним ергономічним вимогам і забезпечувати зручний доступ до основних функцій та операцій системи.

Інтерфейс повинен бути розрахований на переважне використання маніпулятора типу «миша», тобто управління системою має здійснюватися за допомогою набору екранних меню, кнопок, значків і т.п. елементів. Клавіатурний режим введення повинен використовуватися головним чином при заповненні та редагуванні текстових і числових полів екранних форм.

Планується реалізація системи українською мовою. Всі модулі системи, написи екранних форм повинні бути представлені українською мовою, в тому числі і повідомлення видаються користувачеві, за винятком системних повідомлень.

Екранні форми повинні проектуватися з урахуванням вимог уніфікації:

* всі екранні форми користувальницького інтерфейсу повинні бути виконані в єдиному графічному дизайні, з однаковим розташуванням основних елементів керування та навігації;
* для позначення східних операцій повинні використовуватися подібні графічні значки, кнопки та інші керуючі (навігаційні) елементи. Терміни, що використовуються для позначення типових операцій (додавання інформаційної суті, редагування поля даних), а також послідовності дій користувача при їх виконанні, мають бути уніфіковані;
* зовнішня поведінка подібних елементів інтерфейсу (реакція на наведення покажчика «миші», перемикання фокусу, натискання кнопки) будуть реалізовані однаково для однотипних елементів.

У системі повинна бути передбачена авторизація користувачів. Модулі системи повинні бути доступні користувачам у відповідності з рівнем доступу користувача до системи.

Вимоги до надійності програми: надійне (стійке) функціонування програми має бути забезпечене виконанням користувачем сукупності організаційно - технічних заходів, перелік яких наведено нижче:

* організацією безперебійного живлення технічних засобів;
* використанням ліцензійного програмного забезпечення;
* регулярним виконанням рекомендацій про типові норми часу на роботи по сервісному обслуговуванню ПЕОМ та оргтехніки і супроводу програмних засобів;
* регулярним виконанням вимог щодо захисту інформації та випробуваннях програмних засобів на наявність комп'ютерних вірусів.

Час відновлення після відмови , викликаного збоєм електроживлення технічних засобів (іншими зовнішніми факторами), що не є фатальним збоєм (не крах) операційної системи , не повинно перевищувати 30 -ти хвилин за умови дотримання умов експлуатації технічних і програмних засобів. Час відновлення після відмови, викликаного несправністю технічних засобів, фатальним збоєм (крахом) операційної системи, не повинно перевищувати часу, необхідного на усунення несправностей технічних засобів і переустановлення програмних засобів.

Відмови програми можливі внаслідок некоректних дій оператора (користувача) при взаємодії з операційною системою. Щоб уникнути виникнення відмов програми за вказаною вище причини слід забезпечити роботу кінцевого користувача без надання йому адміністративних привілеїв.

Вимоги до кваліфікації та чисельності персоналу: мінімальна кількість персоналу, необхідного для роботи програми, має становити не менше 2 штатних одиниць - адміністратор системи і користувач.

На посаду адміністратора сайту призначається особа, яка відповідає таким вимогам: професійна освіта, стаж роботи у відповідній галузі не менше року.

Посадові обов'язки системного адміністратора

- Конфігурація, настройка і адміністрування персональних комп'ютерів і периферійних пристроїв;

- Підтримка локальної (LAN) мережі;

- Конфігурація, настройка і адміністрування серверів;

- Інсталяція, налаштування, оновлення ОС Windows, системи «АВАС-ДЕНТ», іншого ПЗ (включаючи антивірусні продукти ) ;

- Резервне копіювання і відновлення даних;

- Забезпечення інформаційної безпеки системи;

- Участь у закупівлі обладнання, комплектуючих та програмного забезпечення;

- Ведення технічної документації;

- Консультування користувачів.

Рівень підготовки користувача

Для роботи з програмою користувач повинен володіти навичками роботи з ПК в операційному середовищі Windows.

Кожен користувач у відповідності зі своїми правами повинен володіти необхідними знаннями в предметній області для коректної роботи з системою.

Вимоги до технології зберігання та обробки інформації проектованої завдання та вимоги до системи управління базами даних: з розроблюваної системою повинно мати можливість працювати десять користувачів , тому повинна бути обрана клієнт-серверна технологія зберігання та обробки інформації.

СКБД повинна забезпечувати можливість роботи у багатокористувацькому режимі, відновлення працездатності при програмно - апаратних збоях, таких як відключення електроживлення, проблеми з апаратним забезпеченням, нештатне завершення роботи .

З метою забезпечення надійного функціонування в СКБД має бути передбачено :

* збереження працездатності системи при некоректних операціях користувача (введення некоректних даних);
* збереження цілісності даних при позаштатному завершенні роботи системи;
* резервне копіювання даних;
* журналювання операційної системи.

Даним вимогам відповідають такі СКБД, як, SQL Server, Oracle, InterBase, Firebird. Вибір інших СКБД недоцільний.

Вимоги до складу і параметрів технічних засобів: вимоги до сервера: процесор Intel Core, мінімальний обсяг оперативної пам'яті 2 Гб, мінімальний обсяг дискового простору 60 ГБ, операційна система Windows XP/Vista/7.

Вимоги до клієнтської машині: процесор Intel Core, мінімальний обсяг оперативної пам'яті 2 ГБ, мінімальний обсяг дискового простору 60 ГБ, операційна система Windows XP/Vista/7 та периферійний пристрій – принтер.

Вимоги до вихідних кодів і мов програмування: розробляється система повинна передбачати можливість її подальшого розвитку, модифікації і включення нових функцій в систему, поліпшення коду, можливість розширення механізму аутентифікації.

В якості об`єктно-орієнтованої мови програмування може бути обрані такі як, Delphi, C++, С#, Visual Basic . Вибір інших мов недоцільний.

Вимоги до програмних засобів, які використовуються програмою:

системні програмні засоби, що використовуються програмою, повинні бути представлені ліцензійною версією операційної системи Windows ХР або вище, MS Office, Firebird 2.5 або вище.

Вимоги до організації вхідних та вихідних даних: у процесі роботи програми вхідний інформацією для програми повинні бути: файли баз даних, маніпуляції мишею, а також інформація, що вводиться користувачем на клавіатурі ЕОМ, згідно режимам, що визначаються вихідний екранної інформацією.

Вимоги до захисту інформації та програм: так як інформаційна система буде багатокористувацькою, то в ній будуть використовуватися розмежування доступу до системи, а вхід до системи буде відбуватися за паролем.

Попередній склад програмної документації: склад програмної документації повинен включати наступні документи :

* технічне завдання (постановку завдання);
* керівництво користувача, що містить опис усіх задокументованих можливостей програми;
* текст програми, що містить вихідний код на одній з мов, зазначених у пункті;
* керівництво програміста, що містить дані про технології проектування і програмування даної програми, а також про мову програмування, СКБД , призначення програми ;
* список літератури, що містить теоретичний матеріал, необхідний для створення програми.

# 2. СПЕЦІАЛЬНИЙ РОЗДІЛ

## 2.1 Проектування підсистеми, що розробляється

### 2.1.1 Методи та засоби структурного системного аналізу та проектування

Мета проектної задачі:

* розробка і впровадження автоматизованої інформаційної системи

обліку роботи стоматологічного центру;

* розробка системи, яка допоможе адміністратору стоматологічного центру в обробці, пошуку, зберіганні інформації про пацієнтів стоматологічного центру, розкладі прийомів і роботи лікарів;
* допомога фахівцю - стоматологу у веденні медичної карти пацієнта;
* допомога медсестрі у веденні обліку матеріалів та інструментарію, а також інформації про лікування хвороби пацієнта;
* підвищення ефективності роботи всіх підрозділів стоматологічного центру та ведення обліку в єдиній інформаційній системі.

Моделювання ділових процесів, як правило, виконується за допомогою case-засобів. До таких засобів відносяться BPwin (PLATINUM technology), Silverrun (Silverrun technology), Oracle Designer (Oracle), Rational Rose (Rational Software) і ін. Функціональні можливості інструментальних засобів структурного моделювання ділових процесів будуть розглянуті на прикладі case-засоби BPwin.

BPwin підтримує три методології моделювання: функціональне моделювання (IDEF0); опис бізнес-процесів (IDEF3); діаграми потоків даних (DFD).

BPwin має досить простий і інтуїтивно зрозумілий інтерфейс користувача.

Мета моделювання визначається з відповідей на наступні питання:

* Чому цей процес повинен бути змодельований?
* Що повинна показувати модель?
* Що може отримати клієнт?
* Точка зору (Viewpoint).

Під точкою зору розуміється перспектива, з якою спостерігалася система при побудові моделі. Хоча при побудові моделі враховуються думки різних людей, всі вони повинні дотримуватися єдиної точки зору на модель. Точка зору повинна відповідати меті і кордонів моделювання. Як правило, вибирається точка зору людини, відповідального за моделируемую роботу в цілому.

IDEF0-модель передбачає наявність чітко сформульованої мети, єдиного суб'єкта моделювання та однієї точки зору.

Основу методології IDEF0 складає графічний мова опису бізнес-процесів. Модель в нотації IDEF0 являє собою сукупність ієрархічно впорядкованих і взаємопов'язаних діаграм. Кожна діаграма є одиницею опису системи і розташовується на окремому аркуші.

Модель може містити чотири типи діаграм:

* контекстну діаграму (у кожній моделі може бути тільки одна контекстна діаграма);
* діаграми декомпозиції;
* діаграми дерева вузлів;
* діаграми тільки для експозиції (FEO).

IDEF3 - методологія моделювання, что вікорістовує графічний описание інформаційних потоків, взаємін между процесами ОБРОБКИ ІНФОРМАЦІЇ ТА об'єктів, что є Частинами ціх процесів. IDEF3 дает можлівість аналітікам описати сітуацію, коли процеси віконуються в певній послідовності, а також описати об'єкти, що беруть доля спільно в одному процесі. Будь-яка IDEF3-діаграма може містити роботи, зв'язку, перехрестя и об'єкти посилання.

Діаграми потоків даних (Data flow diagram, DFD) використовуються для опису документообігу та обробки інформації. Подібно IDEF0, DFD представляє систему як мережу пов'язаних між собою робіт. Їх можна використовувати як доповнення до моделі IDEF0 для більш наочного відображення поточних операцій документообігу в корпоративних системах обробки інформації. Головна мета DFD - показати, як кожна робота перетворює вхідні дані у вихідні, а також виявити відносини між цими роботами. Будь-яка DFD-діаграма може містити роботи, зовнішні сутності, стрілки (потоки даних) і сховища даних.

 FEO (For Exposition Only) діаграми (інша назва - діаграми тільки для експозиції, описи) використовуються для ілюстрації альтернативної точки зору, для відображення окремих деталей, які не підтримуються явно синтаксисом IDEF0. FEO діаграми дозволяють порушити будь синтаксичне правило, постільки ці діаграми - фактично звичайні картинки - копії стандартних діаграм.

#### 2.1.1.1 Функціональна модель системи за методологією IDEF0

В якості предметної області виступатиме вигаданий стоматологічний цент який займається виконанням лікувальних та профілактичних заходів сучасної стоматології. Основні процедури в стоматологічному центрі:

* адміністратор веде запис пацієнтів на прийом;
* стоматологи виконують огляд та лікування пацієнтів;
* стоматологи ведуть стоматологічну карту пацієнта;
* складають списки необхідного та передають до медсестр;
* медсестри ведуть облік запасів, та закуповують у постачальників необхідні матеріали та інструменти.

Компанія використовує куплену бухгалтерську інформаційну систему, яка дозволяє оформити замовлення, рахунок і відстежити платежі за рахунками.

Розглянемо особливості розглянутої роботи.

Вхід - це споживана або змінна роботою інформація або матеріал

Вихід - інформація або матеріал, які виробляються роботою

Управління - процедури, правила, стратегії або стандарти, якими керується робота

Механізми - ресурси, які виконують роботу (наприклад, співробітники, обладнання, пристрої )

Для розглянутого підприємства вхідними стрілками будуть:

Прийом пацієнтів - скарги і побажання, які турбують пацієнта;

Матеріали і інструменти від постачальників - матеріали і інструменти , отримані від постачальників, за допомогою яких виконуються медичні послуги.

Вихідні стрілки:

Надані медичні послуги – послуги, які отримав пацієнт, після лікування чи діагностики;

Маркетингові матеріали - прайс-листи;

Замовлення постачальникам - список матеріалів і інструментів, які стомотологічний цетр закуповує у постачальників;

Оплата за матеріали і інструменти - гроші постачальникам за матеріали і інструменти;

Медична карта – реєстр причин та умов протікання хвороб, алергічних реакцій пацієнта на застосовувані препарати.

Стрілки управління:

Законодавство - різні законодавчі документи, якими керується стомотологічний центр в процесі своєї діяльності;

Правила і процедури - різні правила і процедури, якими керується стомотологічний центр в процесі своєї діяльності (правила поведінки з медичним обладнанням, процедура спілкування з клієнтами).

Стрілки механізмів: Бухгалтерська система; Персонал; Медичне обладнання.

Підсумкова контекстна діаграма має вигляд (рис.2.2.1.1):

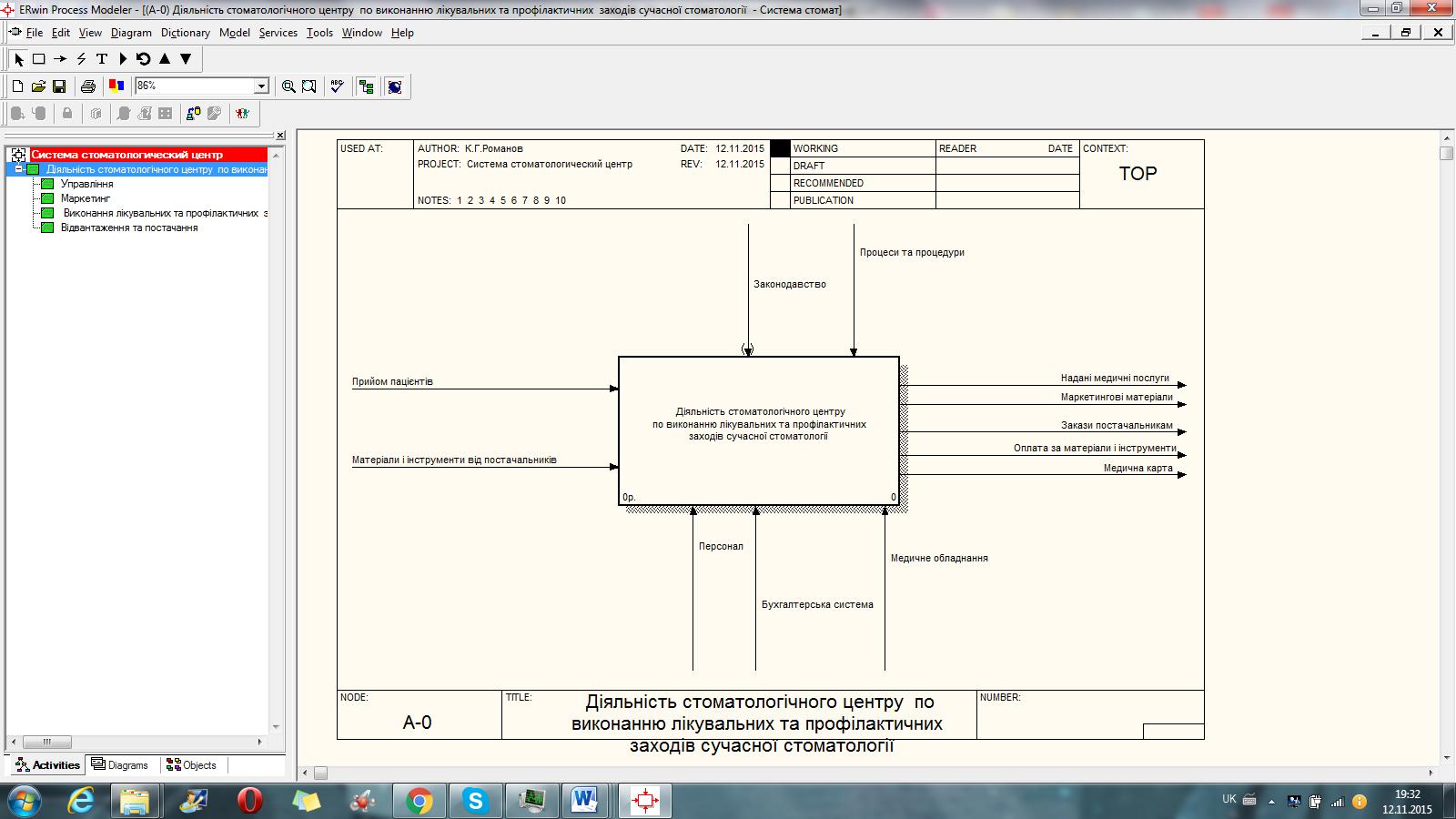


Рисунок 2.2.1.1 - Підсумкова контекстна діаграма

##### Побудова діаграми декомпозиції першого рівня в нотації IDEF0 (модель AS-IS)

У попередньому завданні була побудована контекстна діаграма, що містить тільки одну роботу, яка описує діяльність підприємства в цілому, без деталізації складових цієї роботи. У даній роботі буде побудовані діаграма декомпозиції першого рівня в нотації IDEF0.

За допомогою діаграми декомпозиції першого рівня покажемо, з яких більш дрібних робіт полягає робота. У даній роботі були виділені наступні дочірні роботи (табл.2.2.1.1 ).

Після створення дочірньої діаграми першою дією з'єднаємо граничні стрілки з роботами. Стрілку "Прийом пацієнтів" з'єднаємо з роботою " Виконання лікувальних та профілактичних заходів стоматології ", стрілку " Матеріали і інструменти від постачальників " - з "Відвантаження та постачання". Виходом роботи "Управління" буде " Оплата за матеріали і інструменти ", виходом "Маркетинг" - "Маркетингові матеріали". Стрілки " Надані медичні послуги " і " Медична карта " - вихід роботи " Виконання лікувальних та профілактичних заходів стоматології ".

Таблиця 2.2.1.1– Дочірні роботи

|  |  |
| --- | --- |
| Управління | Дана робота включає в себе загальне управління підприємством, фінансами, кадрами, бухгалтерію . |
| Маркетинг | Презентації, виставки, реклама, маркетингові дослідження. |
| Виконання лікувальних та профілактичних заходів стоматології | Виконанню лікувальних та профілактичних заходів стоматології |
| Відвантаження та постачання | Постачання підприємства необхідними комплектуючими, зберігання та відвантаження готової продукції |

Стрілка "Закази постачальникам - вихід роботи "Відвантаження та постачання".

Стрілка "Персонал" буде бути входом механізму всіх чотирьох робіт, а стрілка "Бухгалтерська система" - робіт "маркетинг" і "Відвантаження та отримання". Стрілка "Правила і процедури" буде входом управління всіх чотирьох робіт. Стрілка " Медичне обладнання" буде входом управління роботи «Виконання лікувальних та профілактичних заходів стоматології».

«Гроші пацієнтів» - це вхід роботи " Діяльність стоматологічного центру по виконанню лікувальних та профілактичних заходів сучасної стоматології ". Представлена діаграма на рис. 2.1.1.1.1.

Після з'єднання граничних стрілок з роботами наступним кроком з'єднаємо роботи між собою. Оскільки робота "Управління" включає в себе загальне управління підприємством, то одним з її результатів буде "Керуюча інформація", що надходить на вхід управління всіх інших робіт.

Роботі "Виконання лікувальних та профілактичних заходів стоматології " для свого функціонування необхідні матеріали та інструменти, які вона замовляє у роботи "Відвантаження та постачання" (вихідна стрілка "Список необхідних матеріалів та інструменти ").

Інформація про результати медичних досліджень та послуг необхідна роботі "Маркетинг" (вихідна стрілка "Результати ").

Результатом роботи "Відвантаження та постачання" будуть необхідні матеріали та інструменти, які надходять на вхід роботи " Виконання лікувальних та профілактичних заходів стоматології ".

Управління будь-якого підприємства має знати, що відбувається на підприємстві, чим займається кожен підрозділ і які результати їх роботи, тобто будь-яка робота в ідеалі повинна звітувати про результати своєї діяльності перед управлінням. Створимо стрілки виходів робіт " Маркетинг ", " Виконання лікувальних та профілактичних заходів стоматології " і "Відвантаження та постачання" і з'єднаємо їх зі входом управління роботи "Управління" – стрілка “ Звіти “.

Модифікуємо діаграму, візуально виділивши деякі стрілки. Підсумкова діаграма декомпозиції показана на рис.2.1.1.1.2:

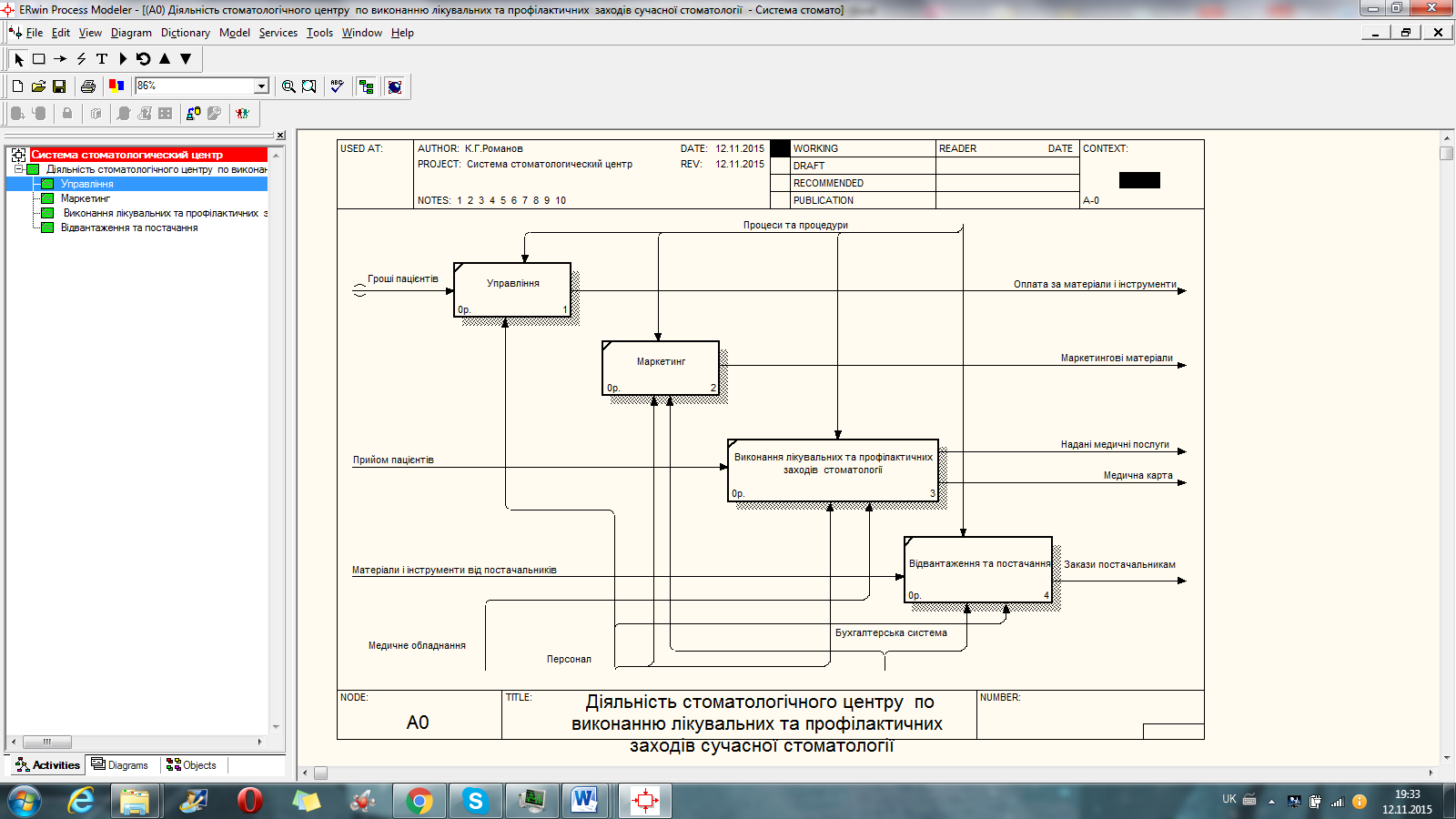


Рисунок 2.1.1.1.1 - Діаграми декомпозиції першого рівня в нотації IDEF0 (модель AS-IS)

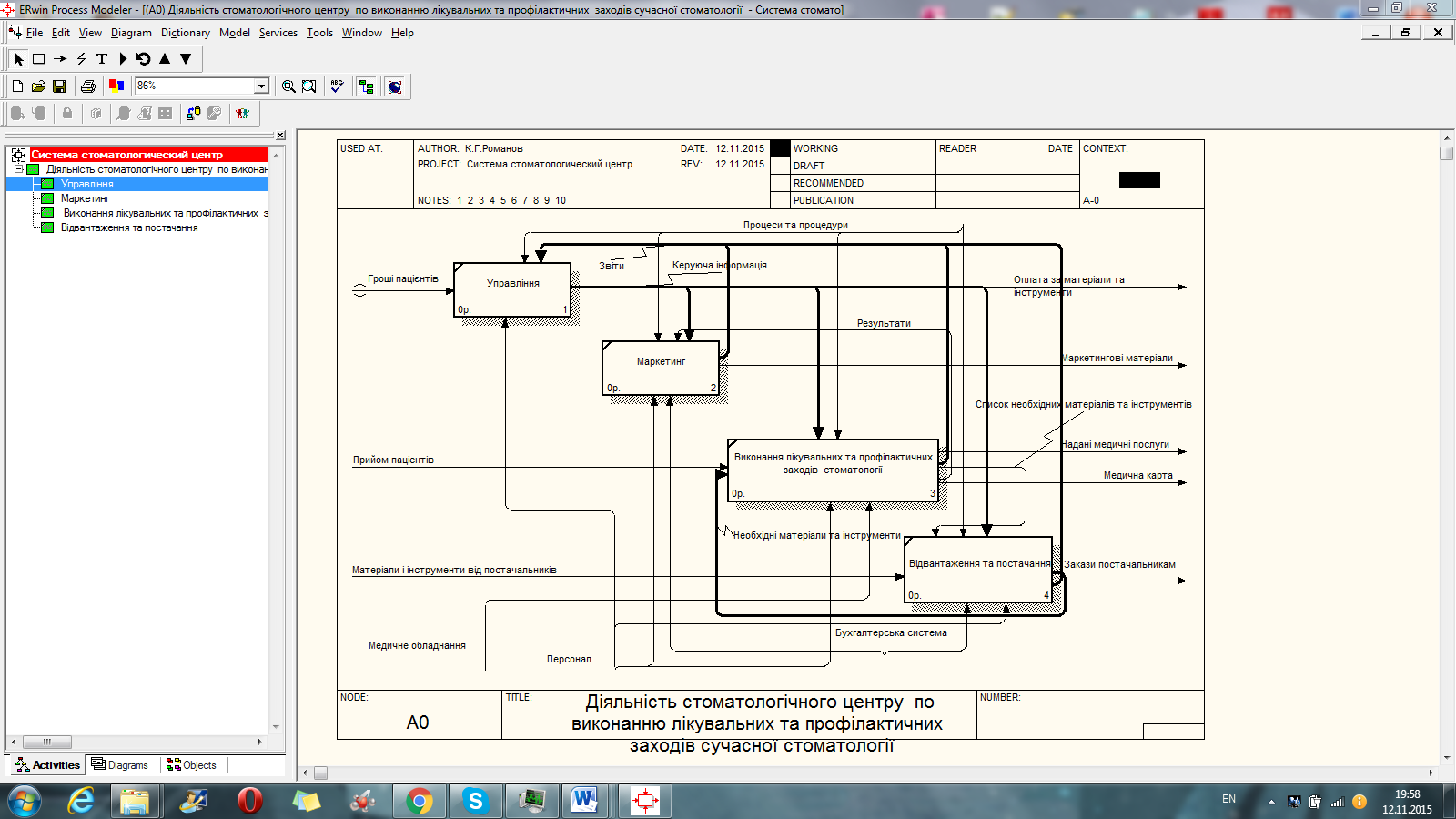


Рисунок 2.1.1.1.2- Підсумкова діаграма декомпозиції

##### Побудова діаграми декомпозиції другого рівня в нотації IDEF0 (модель AS-IS)

У результаті проведення дослідження отримана наступна інформація:

* адміністратор вносить дані про прийом пацієнтів на консультацію, первинний огляд та лікування;
* стоматолог всю інформацію, отриману при огляді пацієнта, вводить у форму медичної карти, для подальшого вирішення;
* стоматолог всю інформацію, отриману при лікуванні пацієнта, вводить у форму медичної карти, для подальшого вирішення;
* після завершення прийому стоматолог подає список необхідних матеріалів, якщо є потреба;
* кожен місяць подаються звіти по прийму та лікуванню пацієнтів;
* лікування та огляд проходить згідно інструкціям та специфікаціями;
* усі результати прийому пацієнта реєструються адміністратором;
* отримані матеріали, використовуються стоматологами в залежності від необхідності лікування пацієнта.

У даній роботі ми виділили чотири дочірні роботи: " Запис пацієнтів на прийом", " Огляд пацієнта", " Виконання лікувальних та профілактичних заходів" і " Ведення медичної карти пацієнта". Як і в попередній роботі почнемо з з'єднання граничних стрілок з роботами.

Стрілка " Необхідні матеріали та інструменти" - це вхід робіт " Огляд пацієнта" і " Виконання лікувальних та профілактичних заходів".

Стрілки управління "Керуюча інформація" з'єднаємо з роботою "Запис пацієнтів на прийом", оскільки саме з цієї роботи починаються всі процеси «Виконання лікувальних та профілактичних заходів стоматології», а стрілку управління "Правила і процедури" - з усіма роботами.

Персонал бере участь у всіх виділених дочірніх роботах, тому заводимо стрілку "Персонал" на вхід механізму всіх робіт (при цьому вказавши, що в першій роботі бере участь адміністратор, а в третій та четвертій стоматолог).

“Список необхідних матеріалів та інструментів” - це один з результатів робіт " Огляд пацієнта" і " Виконання лікувальних та профілактичних заходів". “Результати” - це виходи робіт "Огляд пацієнта", " Виконання лікувальних та профілактичних заходів" і " Ведення медичної карти пацієнта".

Стрілка входу “Прийом пацієнтів” до роботи “Запис пацієнтів на прийом”.

Стрілка виходу “Звіти” з усіх робот. “Медична карта” – вихід “Ведення медичної карти пацієнта”

“Надані медичні послуги” – вихід “Виконання лікувальних та профілактичних заходів” та “Ведення медичної карти пацієнта”.

Після з'єднання граничних стрілок з роботами наступним кроком з'єднаємо роботи між собою.

В залежності від скарг пацієнта, адміністратор записує на прийом і вони надходять на вхід управління робіт " Огляд пацієнта" і " Виконання лікувальних та профілактичних заходів" (стрілки "Запис на огляд і "Запис на лікування", відповідно).

Після огляду, якщо, треба, його результати йдуть до "Виконання лікувальних та профілактичних заходів" (стрілка "Результат огляду") та до роботи "Ведення медичної карти".

стрілка “Результат лікування” – вхід до роботи "Ведення медичної карти пацієнта")

Підсумкова діаграма декомпозиції показана на рис. 2.1.1.1.3:

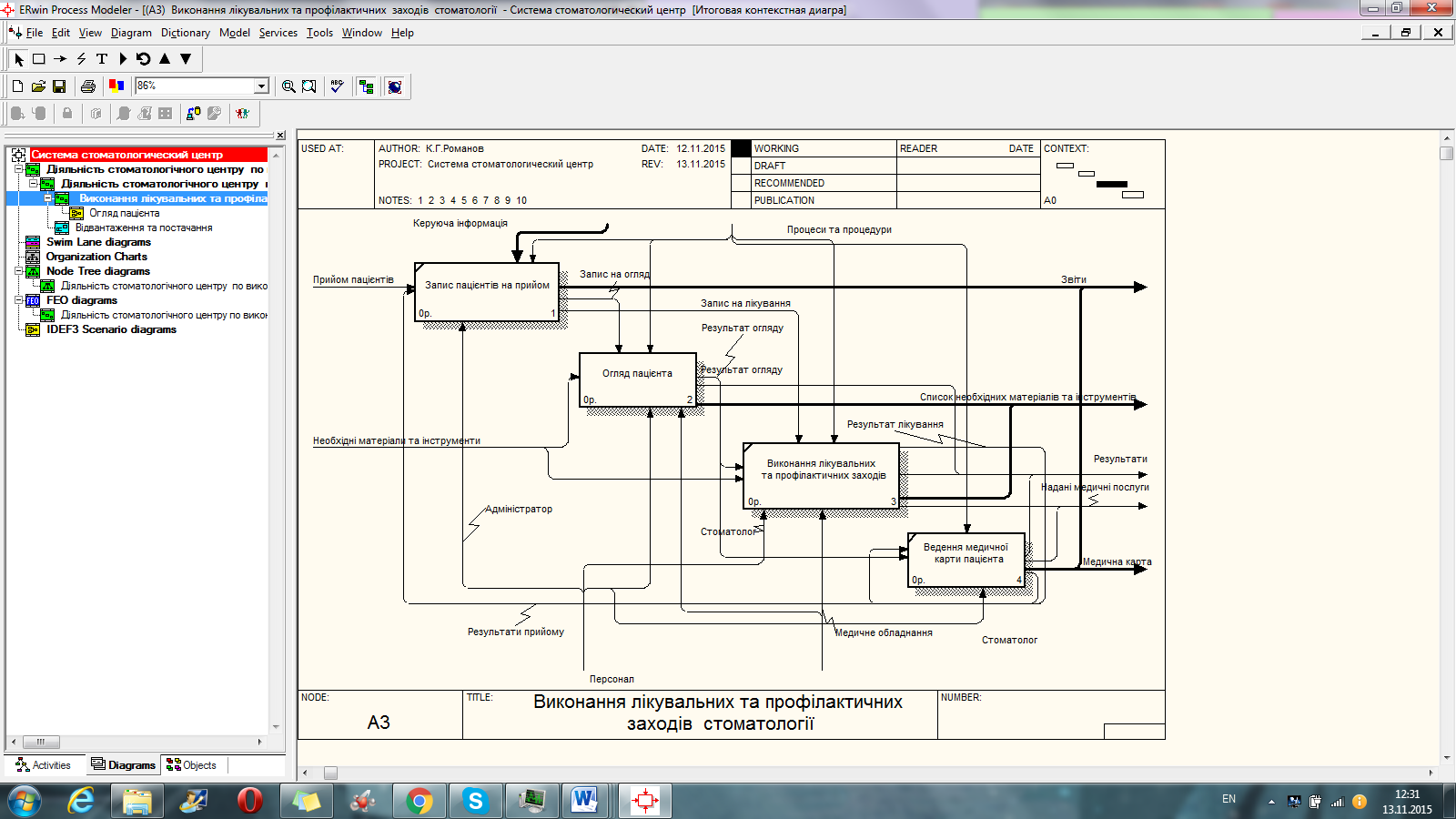


Рисунок 2.1.1.1.3 – Діаграма декомпозиція другого рівня

##### Побудова діаграми декомпозиції першого рівня в нотації IDEF0 (модель TO-BE).

У попередньому завданні була побудована контекстна діаграма, що містить тільки одну роботу, яка описує діяльність підприємства в цілому, без деталізації складових цієї роботи. У даній роботі буде побудовані діаграма декомпозиції першого рівня в нотації IDEF0.

За допомогою діаграми декомпозиції першого рівня покажемо, з яких більш дрібних робіт полягає робота. У даній роботі були виділені наступні дочірні роботи (табл.2.1.1.1.1).

Таблиця 2.1.1.1.1. – Дочірні роботи

|  |  |
| --- | --- |
| Управління | Дана робота включає в себе загальне управління підприємством, фінансами, кадрами, бухгалтерію . |
| Маркетинг | Презентації, виставки, реклама, маркетингові дослідження. |
| Виконання лікувальних та профілактичних заходів | Виконанню лікувальних та профілактичних заходів стоматології |
| Відвантаження та постачання | Постачання підприємства необхідними комплектуючими, зберігання та відвантаження готової продукції |
| Додаток СЦ-ОС | Автоматизація певних процесів роботи |

Після створення дочірньої діаграми першою дією з'єднаємо граничні стрілки з роботами. Стрілку "Прийом пацієнтів" з'єднаємо з роботою " Виконання лікувальних та профілактичних заходів стоматології ", стрілку " Матеріали і інструменти від постачальників " - з "Відвантаження та постачання". Виходом роботи "Управління" буде " Оплата за матеріали і інструменти ", виходом "Маркетинг" - "Маркетингові матеріали". Стрілки " Надані медичні послуги " і " Медична карта " - вихід роботи " Виконання лікувальних та профілактичних заходів стоматології ".

Стрілка "Закази постачальникам - вихід роботи "Відвантаження та постачання".

Стрілка "Персонал" буде бути входом механізму всіх п’яти робіт, а стрілка "Бухгалтерська система" - робіт "маркетинг" і "Відвантаження та отримання". Стрілка "Правила і процедури" буде входом управління всіх чотирьох робіт. Стрілка " Медичне обладнання" буде входом управління роботи «Виконання лікувальних та профілактичних заходів стоматології».

«Гроші пацієнтів» - це вхід роботи " Діяльність стоматологічного центру по виконанню лікувальних та профілактичних заходів сучасної стоматології "

Після з'єднання граничних стрілок з роботами наступним кроком з'єднаємо роботи між собою. Оскільки робота "Управління" включає в себе загальне управління підприємством, то одним з її результатів буде "Керуюча інформація", вона буде надходити на роботу "Додаток СЦ-ОС", а від неї – вихід - надходить на вхід управління всіх інших робіт.

Усі роботи будуть взаємодіяти с роботою "Додаток СЦ-ОС" і вона взаємодіє з усіма іншими роботами.

Роботі "Виконання лікувальних та профілактичних заходів стоматології" для свого функціонування необхідні матеріали та інструменти, які вона замовляє у роботи "Відвантаження та постачання" використовуючи роботу "Додаток СЦ-ОС" (вихідна стрілка "Список необхідних матеріалів та інструменти " на вхід "Додаток СЦ-ОС" – вхід "Відвантаження та постачання").

Інформація про результати медичних досліджень та послуг необхідна роботі "Маркетинг" (вихідна стрілка "Результати " на вхід "Додаток СЦ-ОС" – вхід "Маркетинг" ).

Результатом роботи "Відвантаження та постачання" будуть необхідні матеріали та інструменти, які надходять на вхід роботи " Виконання лікувальних та профілактичних заходів стоматології " через "Додаток СЦ-ОС".

Управління будь-якого підприємства має знати, що відбувається на підприємстві, чим займається кожен підрозділ і які результати їх роботи, тобто будь-яка робота в ідеалі повинна звітувати про результати своєї діяльності перед управлінням. Створимо стрілки виходів робіт " Маркетинг ", " Виконання лікувальних та профілактичних заходів стоматології " і "Відвантаження та постачання" і з'єднаємо їх зі входом через "Додаток СЦ-ОС" , а вона в свою чергу з управлінням роботи "Управління" – стрілка “ Звіти “.

Модифікуємо діаграму, візуально виділивши деякі стрілки. Підсумкова діаграма декомпозиції показана на рис. 2.1.1.1.4:



Рисунок 2.1.1.1.4 - Діаграми декомпозиції першого рівня в нотації IDEF0 (модель TO-BE)

##### Побудова діаграми декомпозиції другого рівня в нотації IDEF0 (модель TO-BE).

У результаті проведення дослідження отримана наступна інформація:

* адміністратор вносить дані про прийом пацієнтів на консультацію, первинний огляд та лікування;
* стоматолог всю інформацію, отриману при огляді пацієнта, вводить у форму медичної карти, для подальшого вирішення;
* стоматолог всю інформацію, отриману при лікуванні пацієнта, вводить у форму медичної карти, для подальшого вирішення;
* після завершення прийому стоматолог подає список необхідних матеріалів, якщо є потреба;
* кожен місяць подаються звіти по прийму та лікуванню пацієнтів;
* лікування та огляд проходить згідно інструкціям та специфікаціями;
* усі результати прийому пацієнта реєструються адміністратором;
* отримані матеріали, використовуються стоматологами в залежності від необхідності лікування пацієнта.

Уся робота проводиться за допомогою додатку СЦ.

У даній роботі ми виділили п’ять дочірніх робот: " Запис пацієнтів на прийом", " Огляд пацієнта", " Виконання лікувальних та профілактичних заходів" і " Ведення медичної карти пацієнта", " Додаток СЦ (адм. та сом.)". Як і в попередній роботі почнемо з з'єднання граничних стрілок з роботами.

Стрілка "Необхідні матеріали та інструменти" - це вхід роботи "Додаток СЦ (адм. та сом.)" яка зв’язується з роботою " Огляд пацієнта" і " Виконання лікувальних та профілактичних заходів".

Стрілки управління "Керуюча інформація" з'єднаємо з роботою "Додаток СЦ (адм. та сом.)" на управління роботи "Запис пацієнтів на прийом", оскільки саме з цієї роботи починаються всі процеси «Виконання лікувальних та профілактичних заходів стоматології», а стрілку управління "Правила і процедури" - з усіма роботами.

Персонал бере участь у всіх виділених дочірніх роботах, тому заводимо стрілку "Персонал" на вхід механізму всіх робіт (при цьому вказавши, що в першій роботі бере участь адміністратор, а в третій та четвертій стоматолог, а в роботі "Додаток СЦ (адм. та сом.)" і стоматолог, і адміністратор).

“Список необхідних матеріалів та інструментів” - це один з результатів робіт " Огляд пацієнта" і " Виконання лікувальних та профілактичних заходів". “Результати” - це виходи робіт "Огляд пацієнта", " Виконання лікувальних та профілактичних заходів" і " Ведення медичної карти пацієнта".

Стрілка входу “Прийом пацієнтів” до роботи “Запис пацієнтів на прийом”.

Стрілка виходу “Звіти” з усіх робот. “Медична карта” – вихід “Ведення медичної карти пацієнта”

“Надані медичні послуги” – вихід “Виконання лікувальних та профілактичних заходів” та “Ведення медичної карти пацієнта”.

Після з'єднання граничних стрілок з роботами наступним кроком з'єднаємо роботи між собою.

В залежності від скарг пацієнта, адміністратор записує на прийом і вони надходять на вхід управління робіт " Огляд пацієнта" і " Виконання лікувальних та профілактичних заходів" (стрілки "Запис на огляд і "Запис на лікування", відповідно через роботу "Додаток СЦ (адм. та сом.)" ).

Після огляду, якщо, треба, його результати йдуть до "Виконання лікувальних та профілактичних заходів" (стрілка "Результат огляду") та до роботи "Ведення медичної карти", усі результати йдуть на вхід "Додаток СЦ (адм. та сом.)" а далі розподіляються в залежності від роботи.

Стрілка “Результат лікування” – вхід до роботи "Додаток СЦ (адм. та сом.)" - на вхід "Ведення медичної карти пацієнта"

Підсумкова діаграма декомпозиції показана на рис. 2.1.1.1.5:

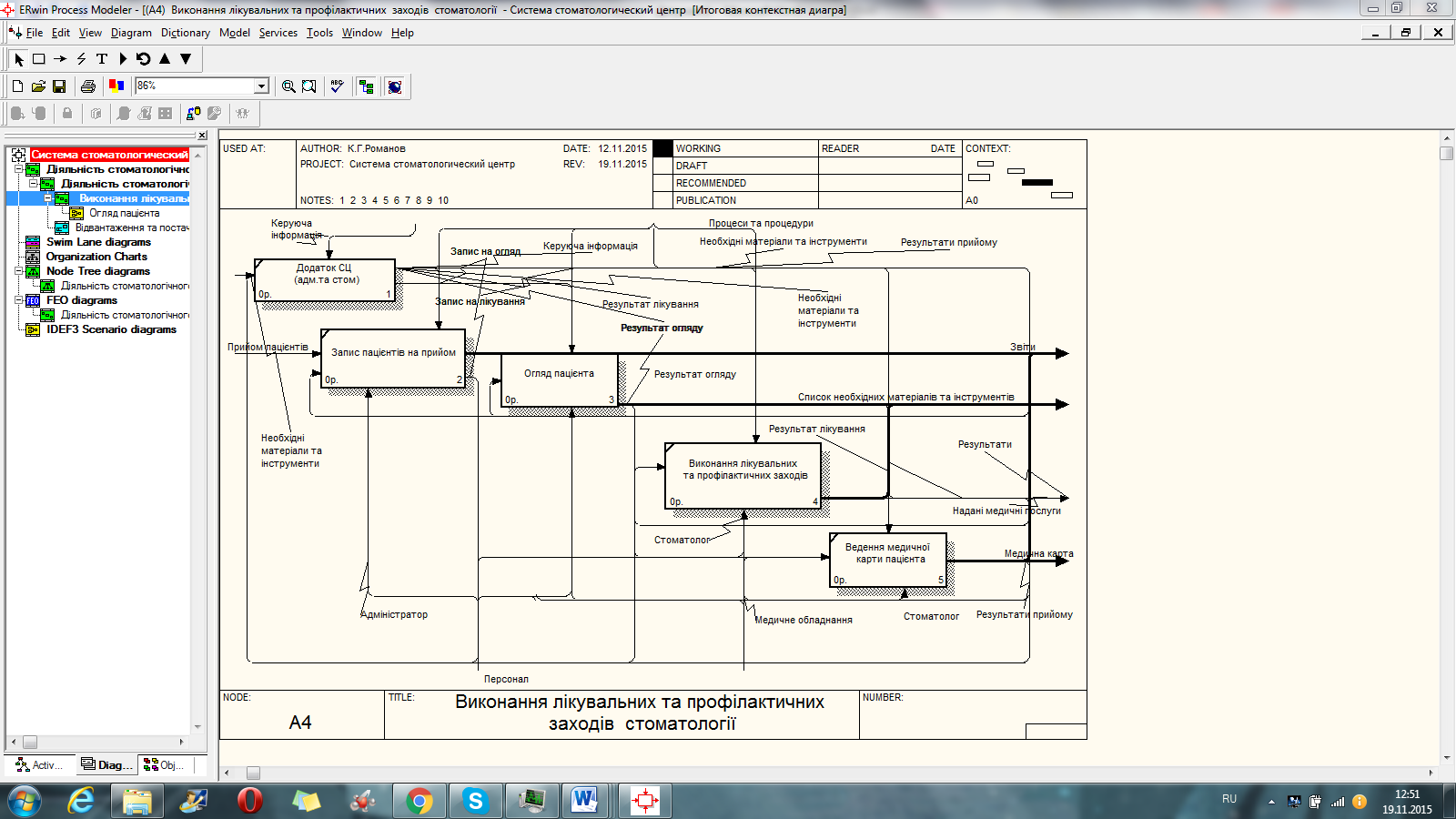


Рисунок 2.1.1.1.5 - Підсумкова діаграма декомпозиції

#### 2.1.1.2 Функціональна модель системи за методологією IDEF3

Проведемо декомпозицію роботи Огляд пацієнта А3 "Виконання лікувальних та профілактичних заходів стоматології ". Дана робота починає виконуватися, коли надходить запис на огляд пацієнта замовлення на збірку. Першою дією буде опитування пацієнта на наявність скарг. Далі підготовлюються інструменти для огляду. Окремо слід виділити заказ необхідних матеріалів та інструментів вразі необхідності.

Наступним кроком починається безпосередньо сам процес огляду порожнини рота пацієнта, в залежності від огляду стоматолог вирішує , що потребує пацієнт: лікування, профілактику чи подальше втручання не вимагається. Після цього наступним кроком йде закінчення огляду.

Розглянемо основні особливості цієї діаграми.

Після опитування пацієнта можливо одне з двох дій - або замовлення зі складу відсутніх матеріалів та інструментів, або, якщо всі інструменті та матеріали в наявності, їх підготовка до огляду. Тому ми поставили перехрестя розгалуження типу "Що виключає АБО". Роботи "Підготовка інструментів для огляду" і "Огляд порожнини рота" з'єднані зв'язком "Потік об'єктів". Тим самим ми показуємо, що між цими роботами передаються об'єкти. Усі наступні роботи з'єднуються зв'язками "старша стрілка", оскільки вони тільки показують послідовність дій над одними і темі ж об'єктами. Після огляду порожнини рота можливе лікування порожнини рота, її профілактика або не втручання , або будь –яка їх комбінація.

Тому ми поставили перехрестя розгалуження типу "Асинхронне АБО". Такий же перехрестя стоїть і після завершення цих робіт.

Далі після "Закінчення огляду" уся інформація фіксується в додатку СЦ, і завдяки цьому буде формуватися "Результати" та "Результат огляду" на (рис. 2.1.1.2).

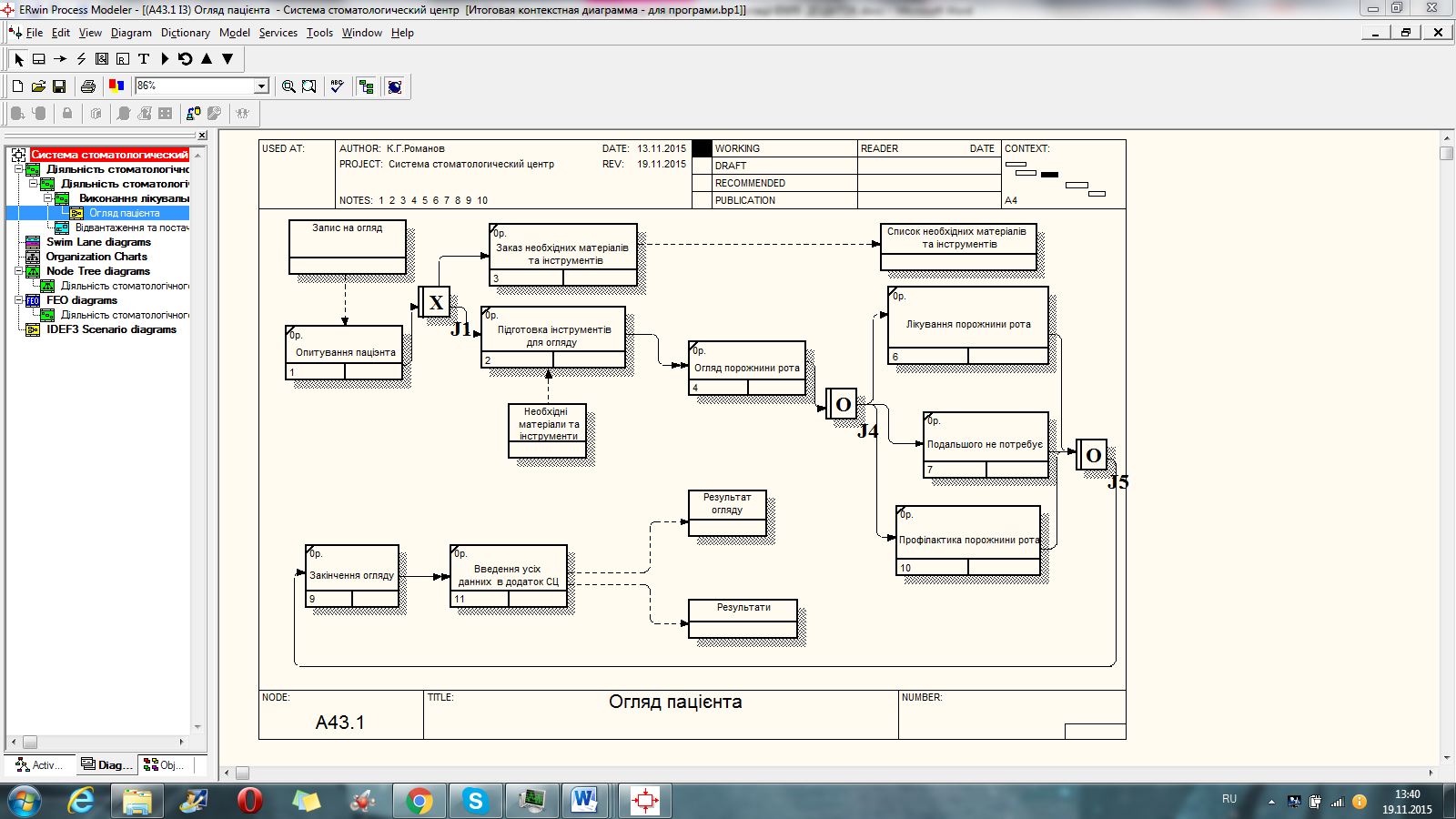


Рисунок 2.1.1.2 - Функціональна модель системи за методологією IDEF3

#### 2.1.1.3 Модель потоків даних

Побудова діаграми декомпозиції. Проведемо декомпозицію роботи Відвантаження та постачання діаграми А0 "Діяльність стоматологічного центу по виконанню лікувальних та профілактичних заходів сучасної стоматології ". У цій роботі ми виділили такі дочірні роботи:

* постачання необхідними матеріалами та інструментами - займається діями, пов'язаними з пошуком відповідних постачальників і замовленням у них необхідних матеріалів і інструментів;
* зберігання запасів матеріалів та інструментів;
* списання матеріалів та інструментів;
* оновлення матеріалів та інструментів.

Центральної тут є робота "Зберігання запасів матеріалів та інструментів ". На її вхід надходять матеріали та інструменти від постачальників, а також список необхідних матеріалів та інструментів, та список оновлень\списань.

Виходом цієї роботи будуть список матеріалів та інструментів, які треба закупити (якщо вони є в наявності), переданий на вхід роботи "Постачання необхідними матеріалами та інструментами ". Виходами робіт " Постачання необхідними матеріалами та інструментами " будуть, відповідно, замовлення постачальникам.

Наступним кроком необхідно визначити, яка інформація необхідна для кожної роботи, тобто необхідно розмістити на діаграмі сховища даних.

Сховища даних є дані взяті з таблиць бази даних додатку СЦ.

Робота " Постачання необхідними матеріалами та інструментами " працює з інформацією про постачальників і з інформацією про замовлення, зроблених у цих постачальників. Стрілка, що з'єднує роботу і сховище даних "База даних додатку Дані список постачальників" двунаправленна, тому робота може як отримувати інформацію про наявних постачальниках, так і вносити дані про нових постачальників. Стрілка, що з'єднує роботу зі сховищем даних " База даних додатку Дані список замовлень" односпрямована, тому робота тільки вносить інформацію про зроблені замовленнях.

Робота " Зберігання запасів матеріалів та інструментів " працює з інформацією про отримувані матеріали та інструменті та ті, які видаються, тому стрілка, що з'єднує роботу з сховищами даних " База даних додатку Дані список матеріалів та інструментів " двонаправлена. Також ця робота при отриманні матеріалів та інструментів повинна робити відмітку про те, що замовлення постачальникам виконане. Для цього вона пов'язана зі сховищем даних " База даних додатку Дані список заказів" односпрямованої стрілкою.

Робота "Списання матеріалів та інструментів " та "Оновлення матеріалів та інструментів " працює з інформацією по списанню та оновленню запасів. . Для цього вводиться відповідні сховища даних, двонаправленна стрілка - " База даних додатку Дані списки оновлень " та " База даних додатку Дані списки списань " на рис 2.1.1.3.

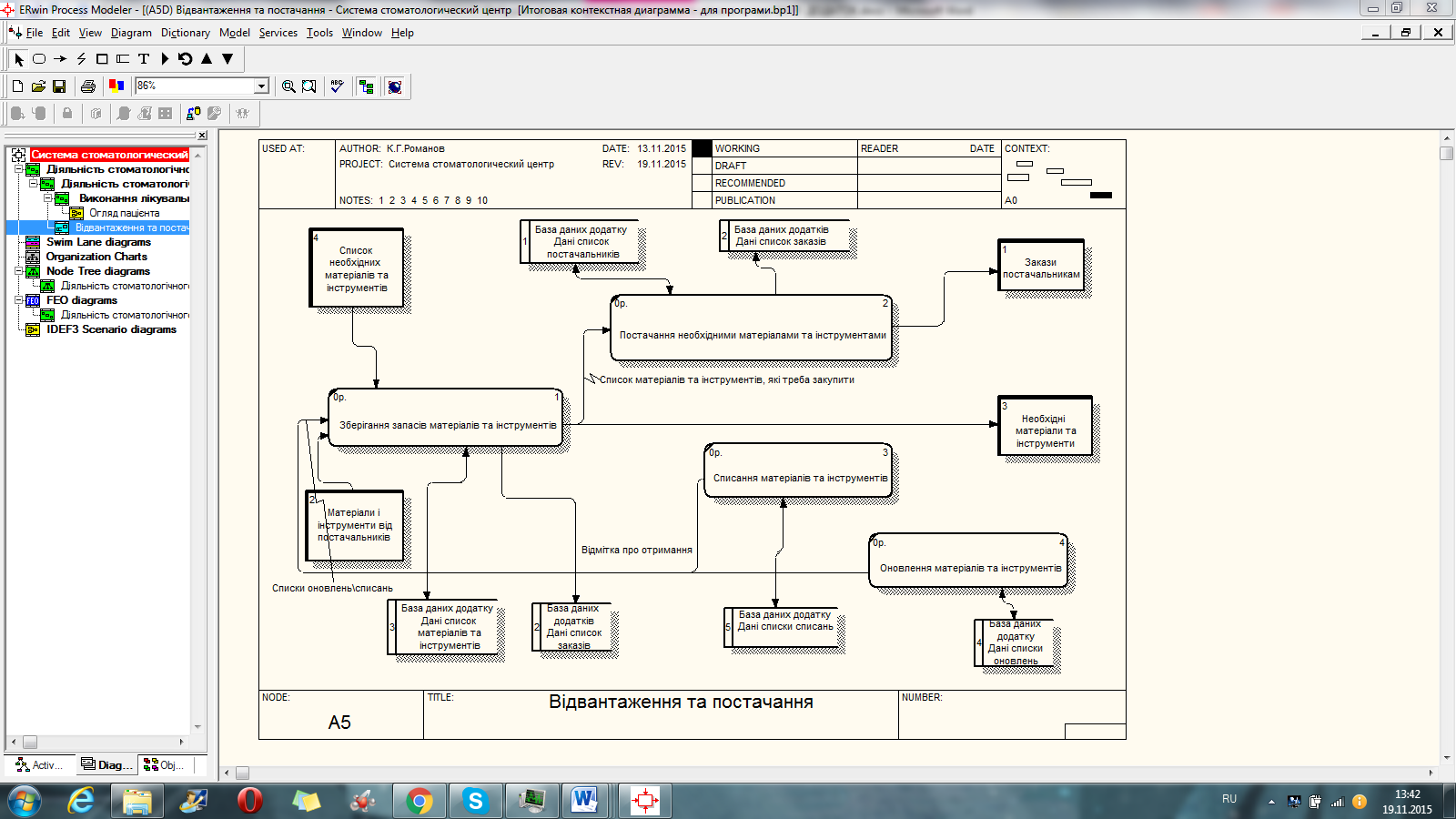


Рисунок 2.1.1.3 - Модель потоків даних

#### 2.1.1.4 Діаграма дерева вузлів

Діаграма дерева вузлів показує ієрархічну залежність робіт, але не взаємозв'язки між роботами. В одній моделі діаграм дерева вузлів може бути безліч, оскільки дерево може бути побудоване на довільну глибину і не обов'язково з кореня. Діаграма дерева вузлів для всіх вузлів моделі показана на рис. 2.1.1.4

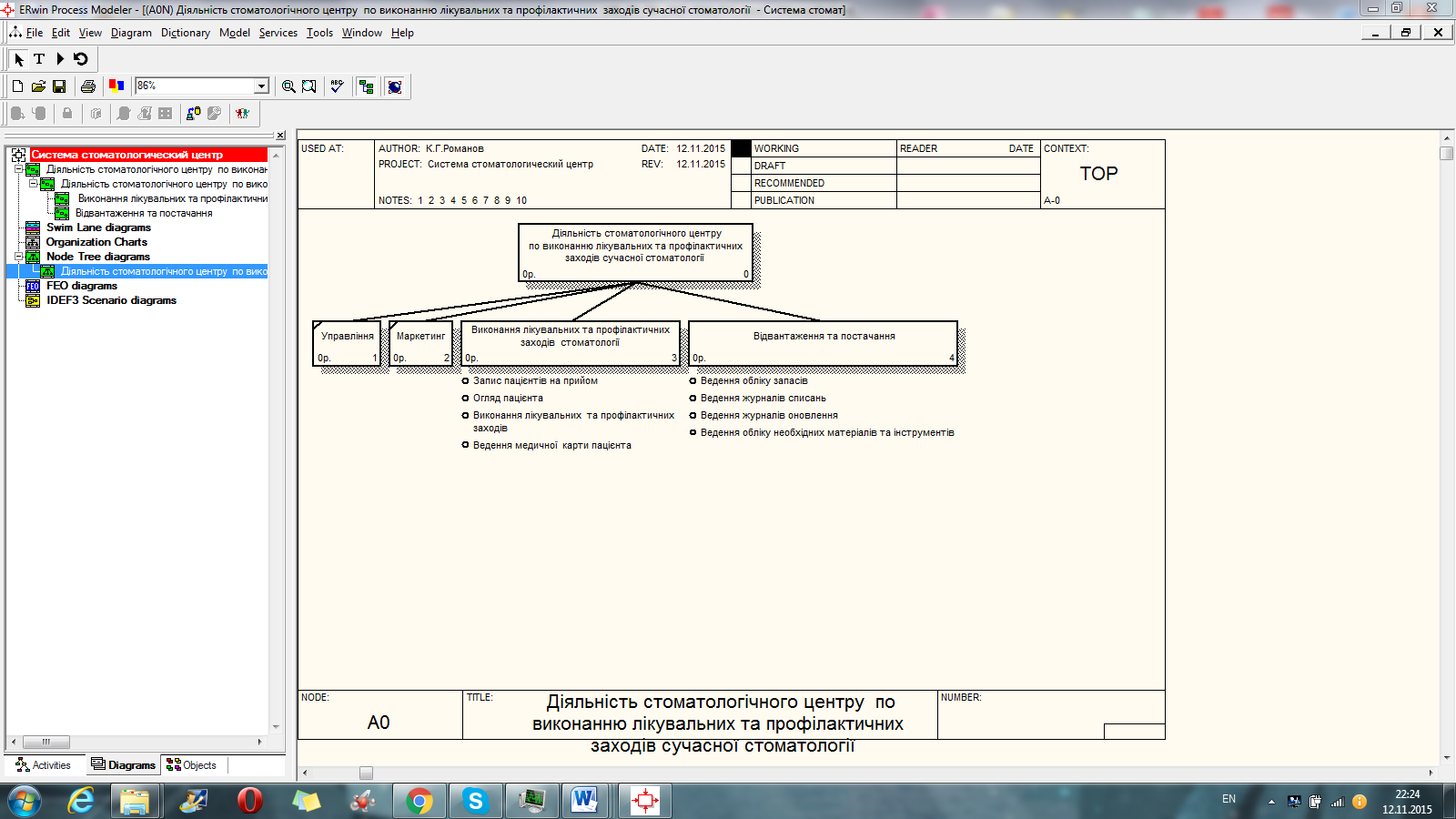


Рисунок 2.1.1.4 - Діаграма дерева вузлів

#### 2.1.1.5 FEO-діаграми

Побудуємо FEO діаграму для діаграми декомпозиції другого рівня А0 "Діяльність стоматологічного центу по виконанню лікувальних та профілактичних заходів сучасної стоматології " і покажемо на ній як дочірні роботи пов'язані між собою. Для цього створюємо діаграму, і видаляємо на ній все граничні стрілки. Підсумкова FEO діаграма показана на рис. 2.1.1.5.



Рисунок 2.1.1.5 - FEO-діаграми

### 2.1.2 Методи та засоби об'єктно-орієнтованого аналізу і проектування

#### 2.1.2.1 Діаграма варіантів використання

На діаграмі представлені актори та прецеденти. Актори поділяються на:

* адміністратор;
* стоматолог;
* медсестра.

Актори будуть проходити авторизацію.

На рисунку 2.1.2.1 зображена діаграма прецедентів для проектованого завдання.

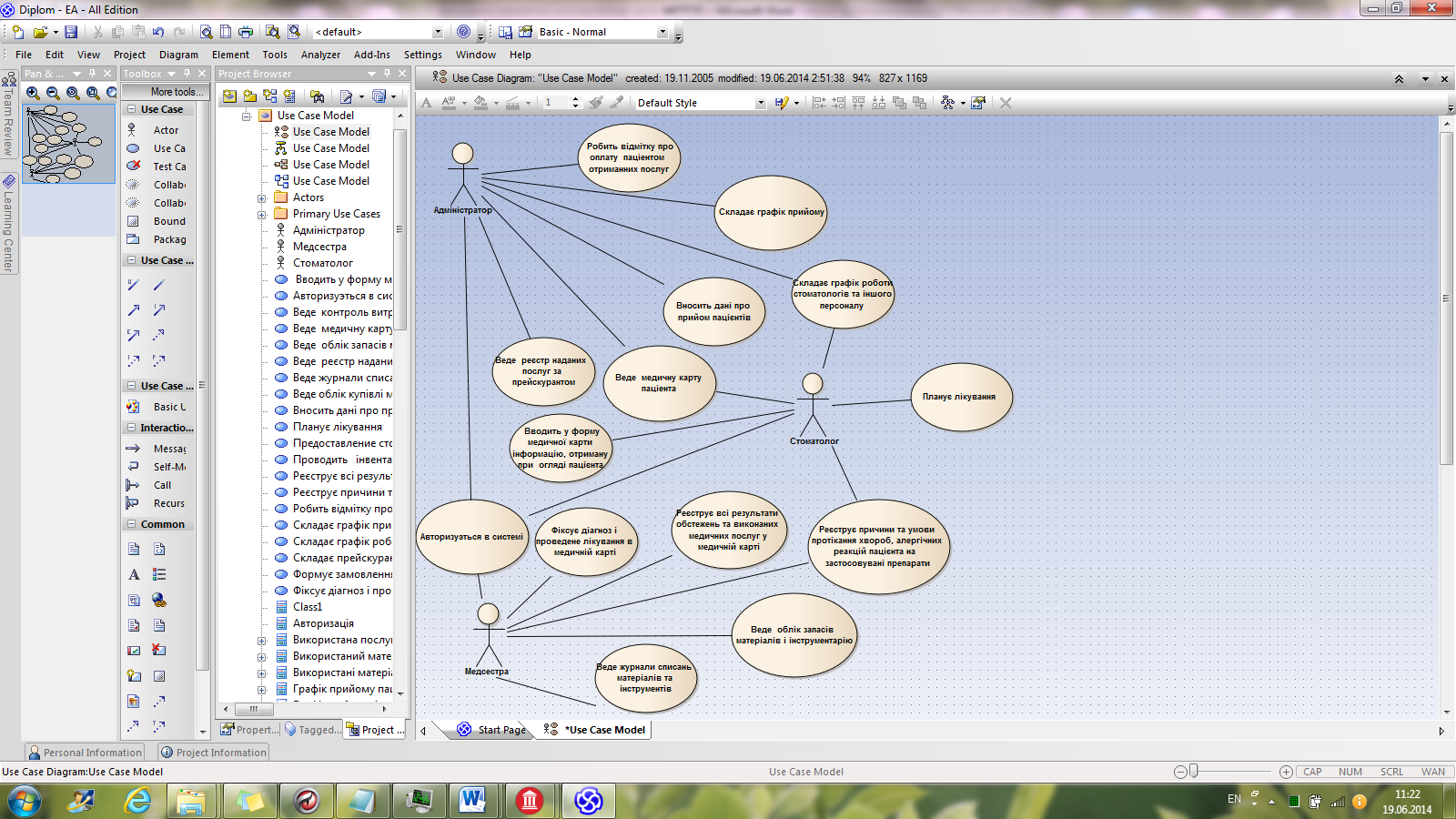


Рисунок 2.1.2.1 - Діаграма прецедентів для проектованого завдання

Адміністратор буде взаємодіяти з такими прецедентами, як :

* авторизується в системі;
* вносить дані про надані послуги;
* веде медичну карту пацієнта;
* вносить дані про прийом пацієнтів;
* складає графік роботи персоналу;

Стоматолог буде взаємодіяти з такими прецедентами, як :

* авторизується в системі;
* веде медичну карту пацієнта;
* веде зубну карту пацієнта;
* складає графік роботи персоналу;
* планує лікування;
* реєструє причини та умови протікання хвороб, алергічних реакцій пацієнта на застосовувані препарати;
* вводить у форму медичної карти інформацію, отриману при огляді пацієнта.

Медсестра буде взаємодіяти з такими прецедентами, як :

* авторизується в системі;
* реєструє всі результати обстежень та виконаних медичних послуг у медичній карті;
* веде журнали онов./спис;
* веде реєстр матеріалів;
* фіксує діагноз і проведене лікування в медичній карті;
* веде контроль витрачання матеріалів;
* реєструє причини та умови протікання хвороб, алергічних реакцій пацієнта на застосовувані препарати.

#### 2.1.2.2 Опис варіанта використання

Варіант використання «Авторизація в системі» в свою чергу зв`язана с усіма акторами: адміністратором, медсестрею, лікарем – стоматологом.

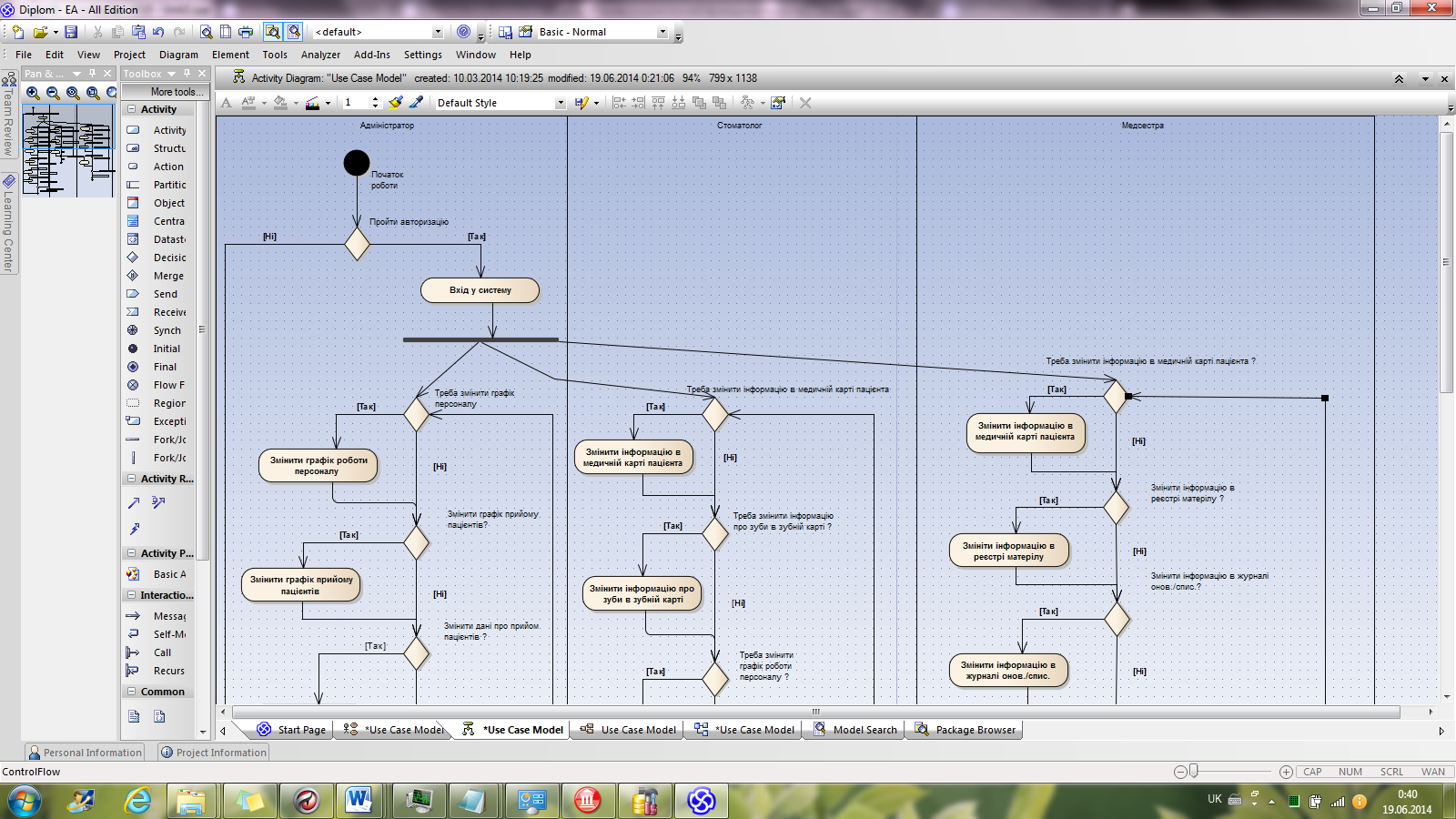
Він має на меті, що при використанні користувачі, актори, будуть проходити авторизацію в системі, для того щоб почати працювати в системі та отримати допуск до необхідної інформації стоматологічного центру.

Він є необхідним для кожного актора, та роботи системи в цілому. З нього починається взаємодія акторів с проектованою системою стоматологічного центру.

#### 2.1.2.3 Діаграма діяльності (activity diagram)

Діаграми діяльності - це один з п'яти видів діаграм, застосовуваних в UML для моделювання динамічних аспектів поведінки системи. Діаграма діяльності - це, по суті, блок - схема, яка показує, як потік управління переходить від однієї діяльності до іншої.

Діаграми діяльності можна використовувати для моделювання динамічних аспектів поведінки системи. Як правило, вони застосовуються, щоб промоделювати послідовні кроки обчислювального процесу. За допомогою діаграм діяльності можна також моделювати життя об'єкта, коли він переходить з одного стану в інший в різних точках потоку управління. Діаграми діяльності можуть використовуватися самостійно для візуалізації, специфікації, конструювання та документування динаміки сукупності об'єктів , але вони придатні також і для моделювання потоку управління при виконанні деякої операції . На рисунку 2.1.2.3 зображена діаграма діяльності проектованої системи.



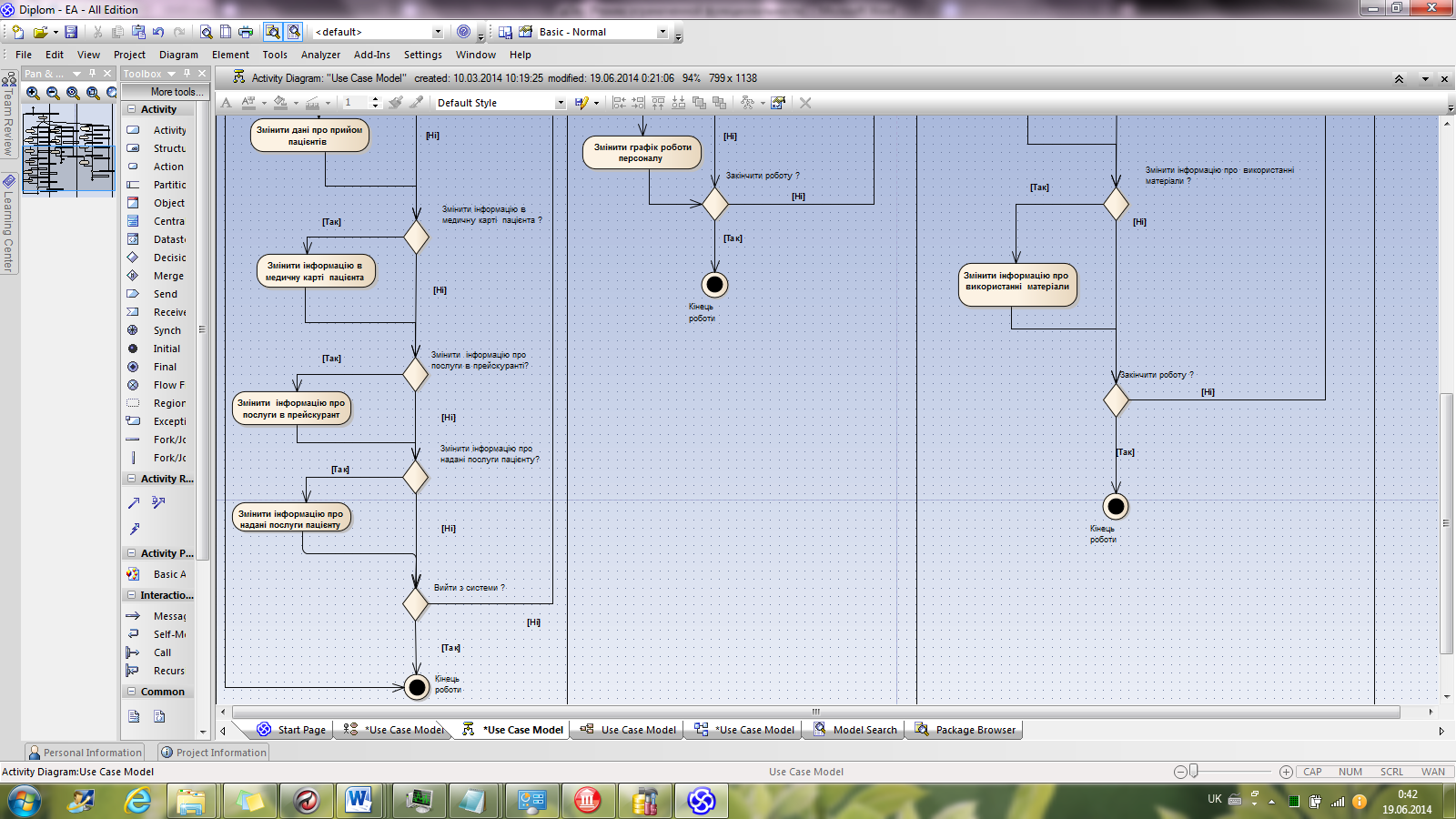


Рисунок 2.1.2.3 - Діаграма діяльності АІМ

Якщо в діаграмах взаємодій акцент робиться на переходах потоку управління від об'єкта до об'єкта , то діаграми діяльності описують переходи від однієї діяльності до іншої. Діяльність - це деякий відносно тривалий етап виконання в автоматі. Зрештою діяльність зводиться до деякого дії , яке складено з атомарних обчислень , що призводять до зміни стану системи або поверненню значення.

Діаграми діяльності важливі не тільки для моделювання динамічних аспектів поведінки системи, але і для побудови виконуваних систем за допомогою прямого і зворотного проектування .

Розроблена система характеризується не тільки структурою складових її елементів , але також і поведінкою ( функціональністю ). При моделюванні поведінки проектованої системи виникає необхідність моделювання логічної реалізації виконуваних системою операцій .

Кожна діаграма діяльності акцентує увагу на послідовності виконання певних дій або елементарних операцій , які в сукупності призводять до отримання бажаного результату. Вони можуть бути побудовані для окремого варіанту використання , кооперації , методу . Застосовувана в них графічна нотація багато в чому схожа на нотацію діаграм станів , оскільки діаграми діяльності є їх різновидом , але якщо на першому основна увага приділяється статичним станам , то на другий - дій.

Кожен стан на діаграмі діяльності відповідає виконанню деякої дії або діяльності , а перехід в наступний стан спрацьовує тільки при їх завершення . В UML дія - це атомарна операція , виконання якої не може бути перервано , а діяльність - неатомарна операція , з можливістю її переривання.

#### 2.1.2.4 Діаграма послідовності взаємодії (Sequence diagram)

Діаграма ж послідовностей відображає взаємодію об'єктів в динаміці. В UML взаємодія об'єктів розуміється як обмін інформацією між ними. При цьому інформація набуває вигляду повідомлень. Крім того, що повідомлення несе якусь інформацію, воно певним чином також впливає на одержувача. Як бачимо, в цьому плані UML повністю відповідає основним принципам ООП, відповідно до яких інформаційну взаємодію між об'єктами зводиться до відправки і прийому повідомлень.

Діаграма послідовностей відноситься до діаграм взаємодії UML, що описує поведінкові аспекти системи, але розглядає взаємодію об'єктів в часі. Іншими словами, діаграма послідовностей відображає тимчасові особливості передачі і прийому повідомлень об'єктами.

Досвідчений читач, можливо, скаже, що щось подібне робить і діаграма прецедентів. Використовувати для уточнення діаграм прецедентів, більш детального опису логіки сценаріїв використання. Це відмінний засіб документування проекту з точки зору сценаріїв використання! Діаграми послідовностей зазвичай містять об'єкти, які взаємодіють в рамках сценарію, повідомлення, якими вони обмінюються, і які повертаються результати, пов'язані з повідомленнями. Втім, часто повертаються результати позначають лише в тому випадку, якщо це не очевидно з контексту.

Тепер про те, які позначення використовуються на діаграмі послідовностей. Як і раніше, об'єкти позначаються прямокутниками з підкресленими іменами (щоб відрізнити їх від класів), повідомлення (виклики методів) - лініями зі стрілками, які повертаються результати - пунктирними лініями зі стрілками. Прямокутники на вертикальних лініях під кожним з об'єктів показують "час життя" (фокус) об'єктів. Втім, досить часто їх не зображують на діаграмі, все це залежить від індивідуального стилю проектування. На рисунку 2.1.2.4 – діаграма послідовності взаємодії.

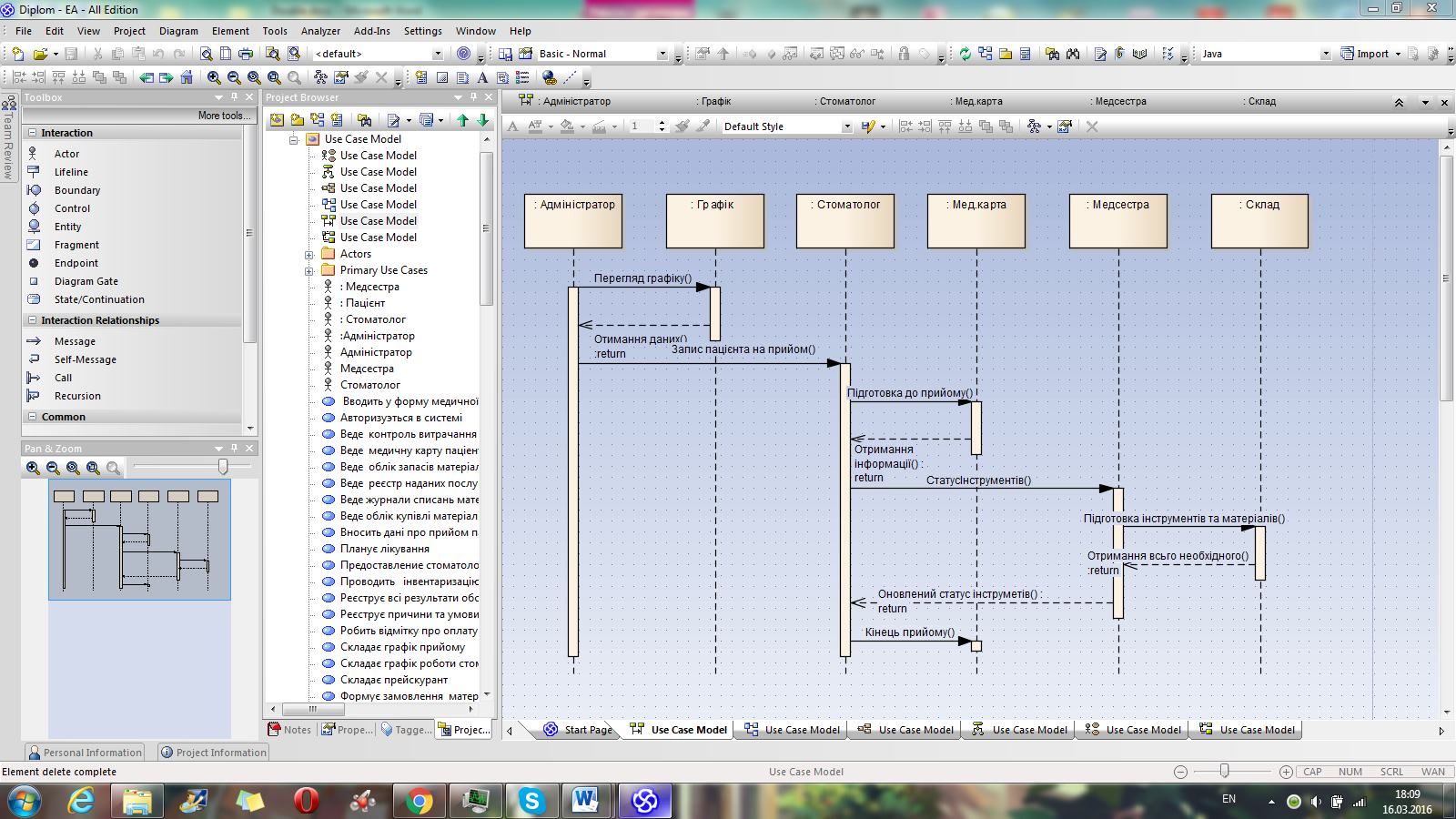


Рисунок 2.1.2.4 – Діаграма послідовності взаємодії

Після приходу пацієнта до стоматології, адміністратор переглядає графік прийому, після чого відбувається запис пацієнта на прийом до стоматолога. Стоматолог ініціює підготовку до прийому, тому перевіряє статус необхідних інструментів. Медсестра підготовлює необхідні інструменти та матеріали взаємодіючі зі складом, отримується результат перевірки і отримання інструментів, та кінець прийому завершується записом в медичну карту.

#### 2.1.2.5  Діаграма класів для вибраного бізнес-варіанта використання

#### 2.1.2.6 Діаграма взаємодії об’єктів (Collaboration diagram) для вибраного бізнес-варіанта використання

Діаграми послідовностей - це відмінний засіб документування поведінки системи, деталізації логіки сценаріїв використання; але є ще один спосіб - використовувати діаграми взаємодії. Діаграма взаємодії показує потік повідомлень між об'єктами системи і основні асоціації між ними і по суті, як уже було сказано вище, є альтернативою діаграми послідовностей. Уважний читач, можливо, скаже, що діаграма об'єктів робить те ж саме, - і не матиме рації. Діаграма об'єктів показує статику, якийсь знімок системи, зв'язки між об'єктами в даний момент часу, діаграма ж взаємодії, як і діаграма послідовностей, показує взаємодію об'єктів у часі, т. е. в динаміці.

Слід зазначити, що використання діаграми послідовностей або діаграми взаємодії - особистий вибір кожного проектувальника і залежить від індивідуального стилю проектування. Ми, наприклад, частіше віддаємо перевагу діаграмі послідовностей. На позначеннях, що застосовуються на діаграмі взаємодії, думаємо, не варто зупинятися докладно. Тут все стандартно: об'єкти позначаються прямокутниками з підкресленими іменами, Асоціації між об'єктами вказуються у вигляді з'єднують їх ліній, над ними може бути зображена стрілка із зазначенням назви повідомлення і його порядкового номера.

Необхідність номера повідомлення пояснюється дуже просто - на відміну від діаграми послідовностей, час на діаграмі взаємодії не відображається у вигляді окремого виміру. Тому послідовність передачі повідомлень можна вказати тільки за допомогою їх нумерації. В цьому і полягає ймовірна причина нехтування цим видом діаграм багатьма проектувальниками. На рисунку 2.1.2.6 – діаграма взамодії.

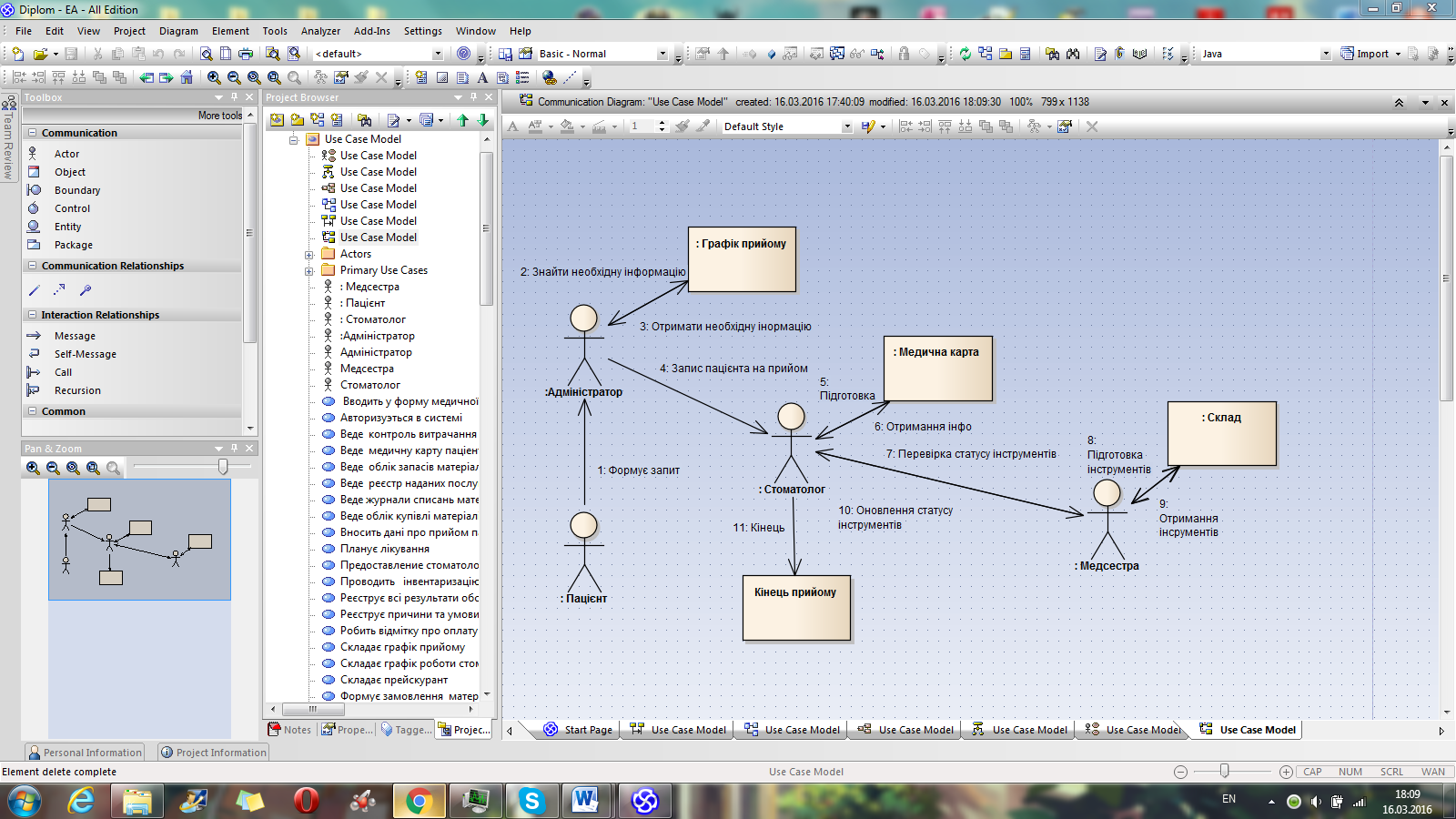


Рисунок 2.1.2.6 – Діаграма взаємодії

#### 2.1.2.7 Діаграма станів і переходів (state transition diagram) для одного з об’єктів

Об'єкти характеризуються поведінкою і станом, в якому знаходяться. Наприклад, людина може бути новонародженим, немовлям, дитиною, підлітком чи дорослим. Іншими словами, об'єкти щось роблять і щось "знають". Діаграми станів застосовуються для того, щоб пояснити, яким чином працюють складні об'єкти. Незважаючи на те що сенс поняття "стан" інтуїтивно зрозумілий, все ж наведемо його визначення в такому вигляді, в якому його дають класики і Zicom Mentor:

Стан (state) - ситуація в життєвому циклі об'єкта, під час якої він задовольняє деякому умовою, виконує певну діяльність або очікує якоїсь події. Стан об'єкта визначається значеннями деяких його атрибутів і присутністю або відсутністю зв'язків з іншими об'єктами.

Діаграма станів показує, як об'єкт переходить з одного стану в інший. Очевидно, що діаграми станів служать для моделювання динамічних аспектів системи (як і діаграми послідовностей, кооперації, прецедентів і, як ми побачимо далі, діаграми діяльності). Часто можна почути, що діаграма станів показує автомат, але про це ми поговоримо докладніше трохи пізніше. Діаграма станів корисна при моделюванні життєвого циклу об'єкта (як і її приватна різновид - діаграма діяльності, про яку ми будемо говорити далі).

Від інших діаграм діаграма станів відрізняється тим, що описує процес зміни станів тільки одного примірника певного класу - одного об'єкта, причому об'єкту реактивного, тобто об'єкта, поведінка якого характеризується його реакцією на зовнішні події. Поняття життєвого циклу може бути застосовано як раз до реактивним об'єктів, даний стан (і поведінку) яких обумовлено їх минулим станом. Але діаграми станів важливі не тільки для опису динаміки окремого об'єкта. Вони можуть використовуватися для конструювання виконуваних систем шляхом прямого і зворотного проектування. І вони дійсно з успіхом застосовуються в такій якості, згадаємо існуючі варіанти "виконуваного UML", такі як UNIMOD, FLORA. На рисунку 2.1.2.7 представлена – діаграма станів і переходів.

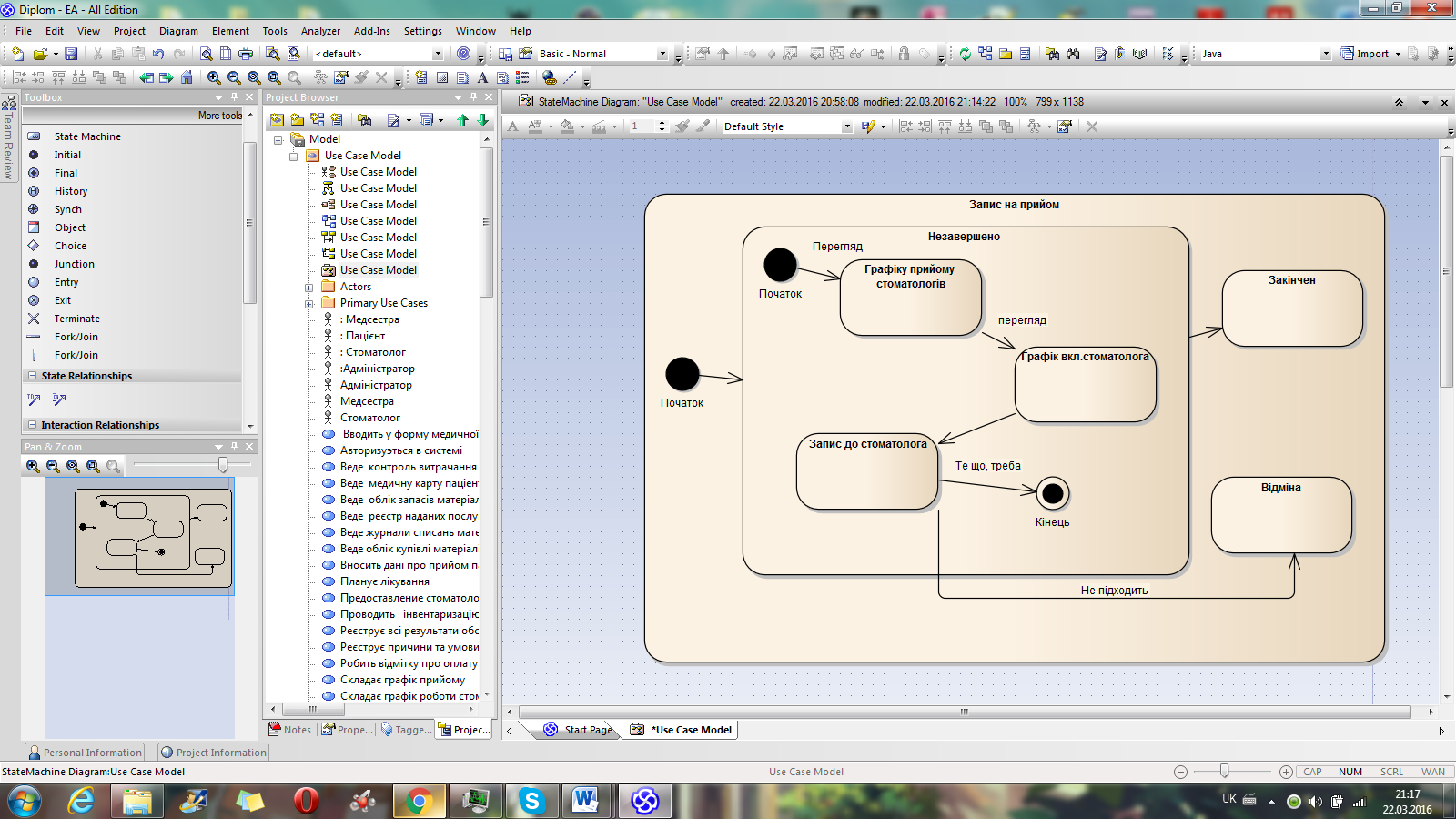


Рисунок 2.1.2.7 - Діаграма станів і переходів (state transition diagram) для одного з об’єктів.

Але поговоримо про позначеннях на діаграмах станів. Округлені прямокутники представляють стану, через які проходить об'єкт протягом свого життєвого циклу. Стрілками показуються переходи між станами, які викликані виконанням методів описуваного діаграмою об'єкта. Існує також два види псевдо стану: початкова, в якому знаходиться об'єкт відразу після його створення (позначається суцільним кружком), і кінцеве, яке об'єкт не може покинути, якщо перейшов в нього (позначається кружком, обведеним колом).

На діаграмі – запис на прийом до лікаря, для цього треба переглянути графік прийому стоматологів, вибрати стоматолога та отримати запис до вибраного стоматолога чи відмінити цей запис.

## 2.2 Інформаційне забезпечення

IDEF1X є методом для розробки реляційних баз даних і використовує умовний синтаксис, спеціально розроблений для зручного побудови концептуальної схеми. Концептуальною схемою ми називаємо універсальне уявлення структури даних в рамках комерційного підприємства, незалежне від кінцевої реалізації бази даних і апаратної платформи. Будучи статичним методом розробки, IDEF1X спочатку не призначений для динамічного аналізу за принципом "AS IS", тим не менш, він іноді застосовується в цій якості, як альтернатива методу IDEF1. Використання методу IDEF1X найбільш доцільно для побудови логічної структури бази даних після того, як всі інформаційні ресурси досліджені (скажімо з допомогою методу IDEF1) і рішення про впровадження реляційної бази даних, як частини корпоративної інформаційної системи, було прийнято. Однак не варто забувати, що засоби моделювання IDEF1X спеціально розроблені для побудови реляційних інформаційних систем, і якщо існує необхідність проектування іншої системи, скажімо об'єктно-орієнтованої, то краще обрати інші методи моделювання.

Основною перевагою IDEF1X, в порівнянні з іншими численними методами розробки реляційних баз даних, такими як ER і ENALIM є жорстка і сувора стандартизація моделювання. Встановлені стандарти дозволяють уникнути різного трактування побудованої моделі, яка безсумнівно є значним недоліком ER.

Розроблено логічне та фізичну модель БД. Логічна модель розширює модель шляхом визначення для сутностей їх атрибутів, описів і обмежень, уточнює склад сутностей і взаємозв'язки між ними.

На діаграмі зображено десть сутностей, які взаємодіють між собою.

- медична карта;

- зубна карта;

- персонал;

- прайс;

- графік прийому пацієнтів;

- графік роботи лікарів;

- використанні послуги;

- реєстр матеріалів;

- використанні матеріали;

- журнал списань, оновлень.

На рисунку зображена діаграма логічної моделі рис (2.2)

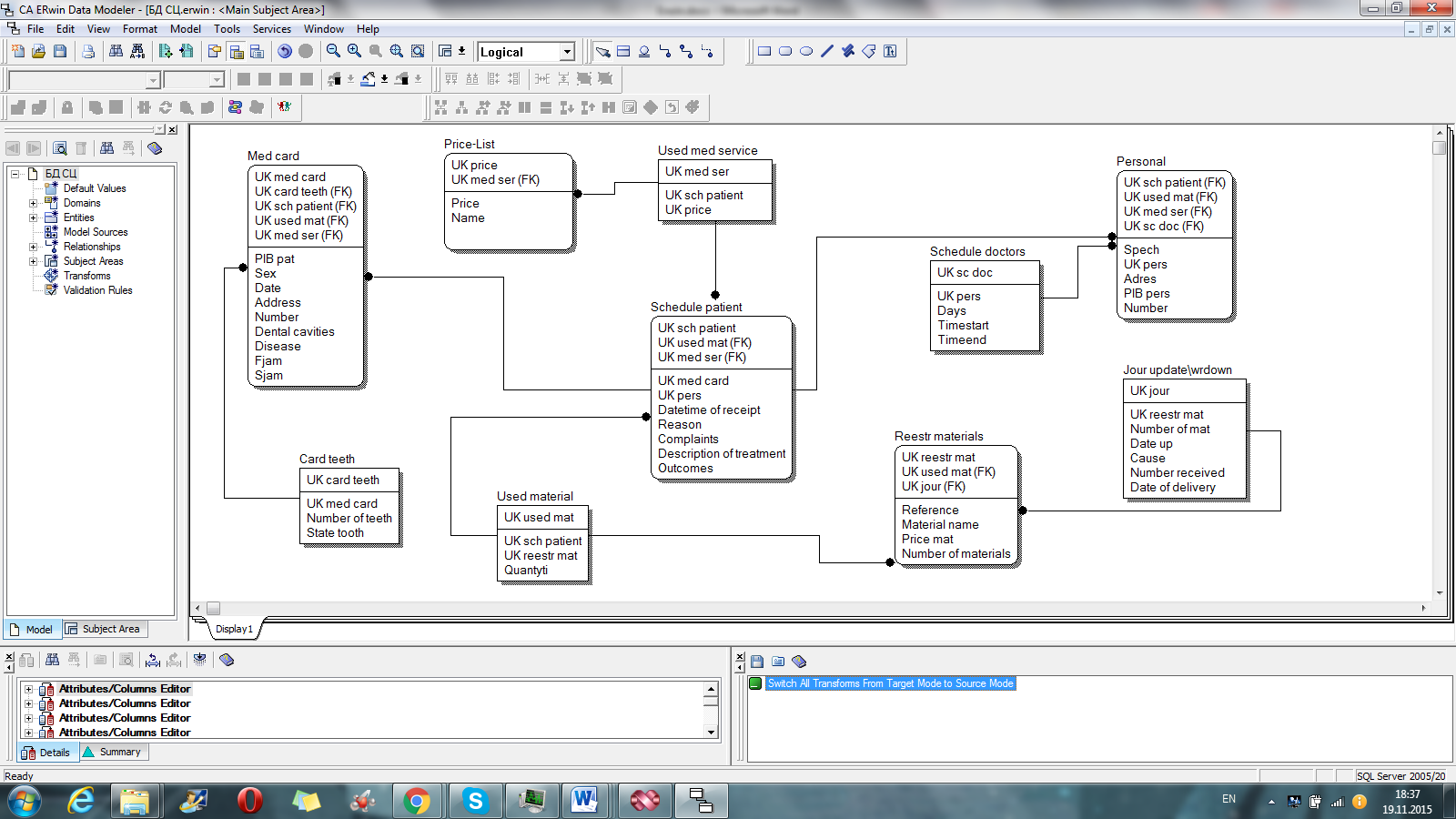


Рисунок 2.2 - Логічна модель БД

Фізична модель даних описує реалізацію об'єктів логічної моделі на рівні об'єктів конкретної бази даних.

А фізичну на рис. 2.2 (1).

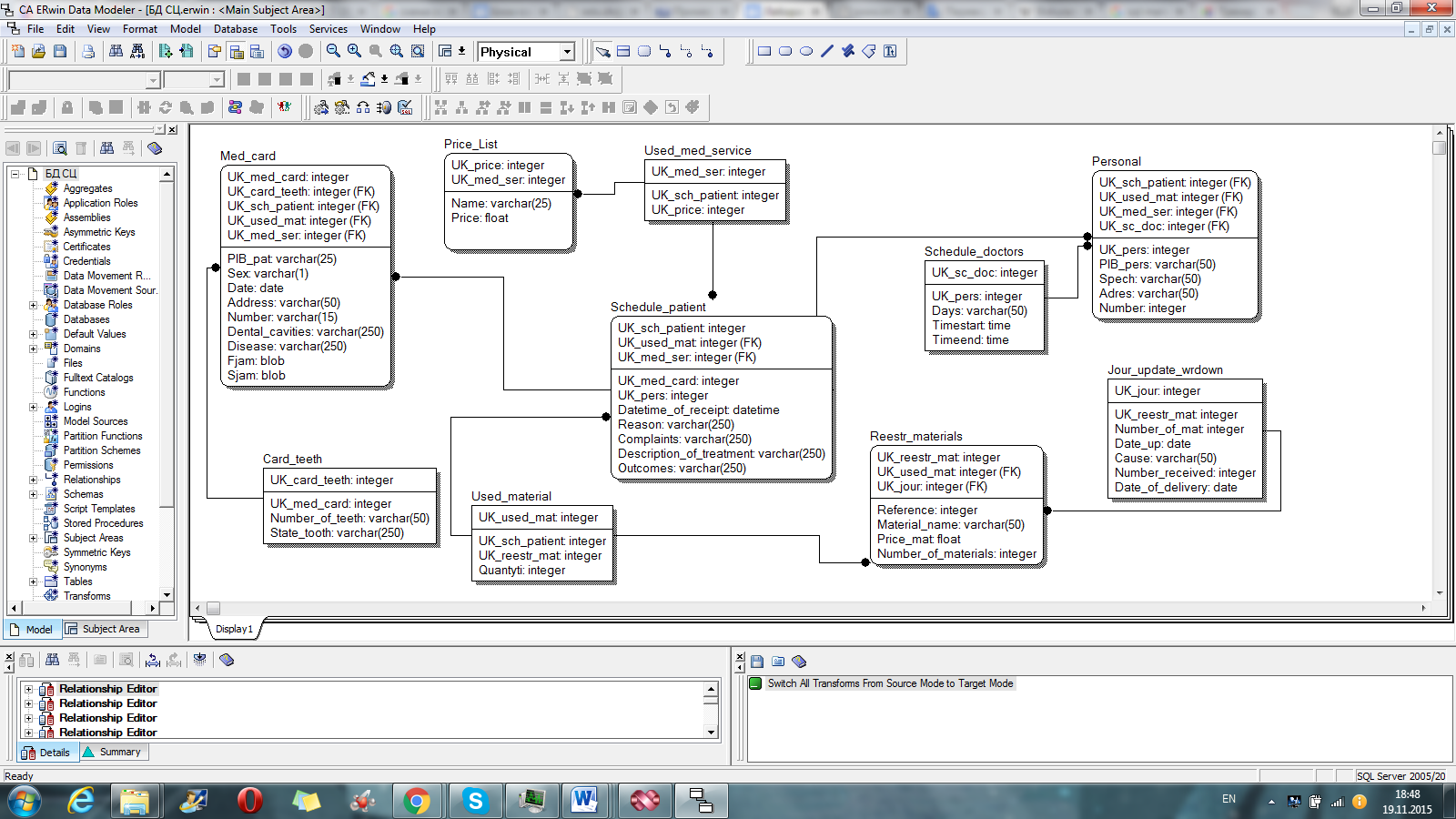


Рисунок. 2.2.(1) - Фізична модель БД

### 2.2.1 Концептуальна модель АІС

Концептуальна модель розширює модель шляхом визначення для сутностей їх атрибутів, описів і обмежень, уточнює склад сутностей і взаємозв'язки між ними.

На діаграмі зображено десять сутностей, які взаємодіють між собою.

- медична карта;

- зубна карта;

- персонал;

- прайс;

- графік прийому пацієнтів;

- графік роботи лікарів;

- використанні послуги;

- реєстр матеріалів;

- використанні матеріали;

- журнал списань, оновлень.

Сукупність < Використана послуга > представляє собою послуги, які були надані пацієнту. Ключовим атрибутом, який дає доступ до об'єкта < : Використана послуга > , є унікальний атрибут < УК використана послуга > . Сукупність < Використана послуга > асоційований з сукупністю < Графік прийому пацієнта > за допомогою атрибута <УК графік прийому пацієнта >, з сукупністю <Прейскурант> за допомогою атрибута <УК прейскурант >.

Сукупність < Графік роботи лікаря> представляє собою графік роботи лікарів стоматологічного центру. Ключовим атрибутом , який дає доступ до об'єкта < : Графік роботи лікаря > , є унікальний атрибут < УК Графік роботи лікаря >. Сукупність < Графік роботи лікаря > асоційований з сукупністю <Персонал > за допомогою атрибута < УК персонал > .

Сукупність < Персонал> представляє собою персонал стоматологічного центру. Ключовим атрибутом , який дає доступ до об'єкта <: Персонал>, є унікальний атрибут <УК персонал>. Сукупність < Персонал > асоційований з сукупністю <Персонал > за допомогою атрибута < УК персонал > .

Сукупність < Медична карта пацієнта> представляє собою медичну карту пацієнта. Ключовим атрибутом , який дає доступ до об'єкта <: Медична карта пацієнта >, є унікальний атрибут <УК медична карта пацієнта >. Сукупність < Медична карта пацієнта > асоційований з сукупністю <Графік прийому пацієнта > за допомогою атрибута < УК медична карта пацієнта > ; з сукупністю < Зубна карта> за допомогою атрибута < УК медична карта пацієнта>.

Сукупність <Графік прийому пацієнта > представляє собою графік прийому пацієнтів стоматологічного центру. Ключовим атрибутом , який дає доступ до об'єкта <: Графік прийому пацієнта >, є унікальний атрибут <УК графік прийому пацієнта >. Сукупність <Графік прийому пацієнта > асоційований з сукупністю < Медична карта пацієнта > за допомогою атрибута < УК медична карта пацієнта > та з сукупністю < Персонал > за допомогою атрибута < УК персонал >, з сукупністю < Використана послуга> за допомогою атрибута < УК графік прийому пацієнта >, з сукупністю < Використаний матеріал> за допомогою атрибута < УК графік прийому пацієнта > .

Сукупність <Прейскурант> представляє собою прейскурант послуг стоматологічного центру. Ключовим атрибутом , який дає доступ до об'єкта <: Прейскурант >, є унікальний атрибут <УК прейскурант >. Сукупність < Прейскурант > асоційований з сукупністю <Використана послуга > за допомогою атрибута < УК прейскурант > .

Сукупність < Використаний матеріал > представляє собою матеріали, які були використані на одного пацієнта. Ключовим атрибутом , який дає доступ до об'єкта <: Використаний матеріал >, є унікальний атрибут <УК використаний матеріал >. Сукупність < Використаний матеріал > асоційований з сукупністю < Реєстр матеріалу > за допомогою атрибута < УК реєстр матеріалу > , з сукупністю < Графік прийому пацієнта > за допомогою атрибута < УК графік прийому пацієнта >.

На рисунку зображена діаграма концептуальної моделі рис. (2.2.1)

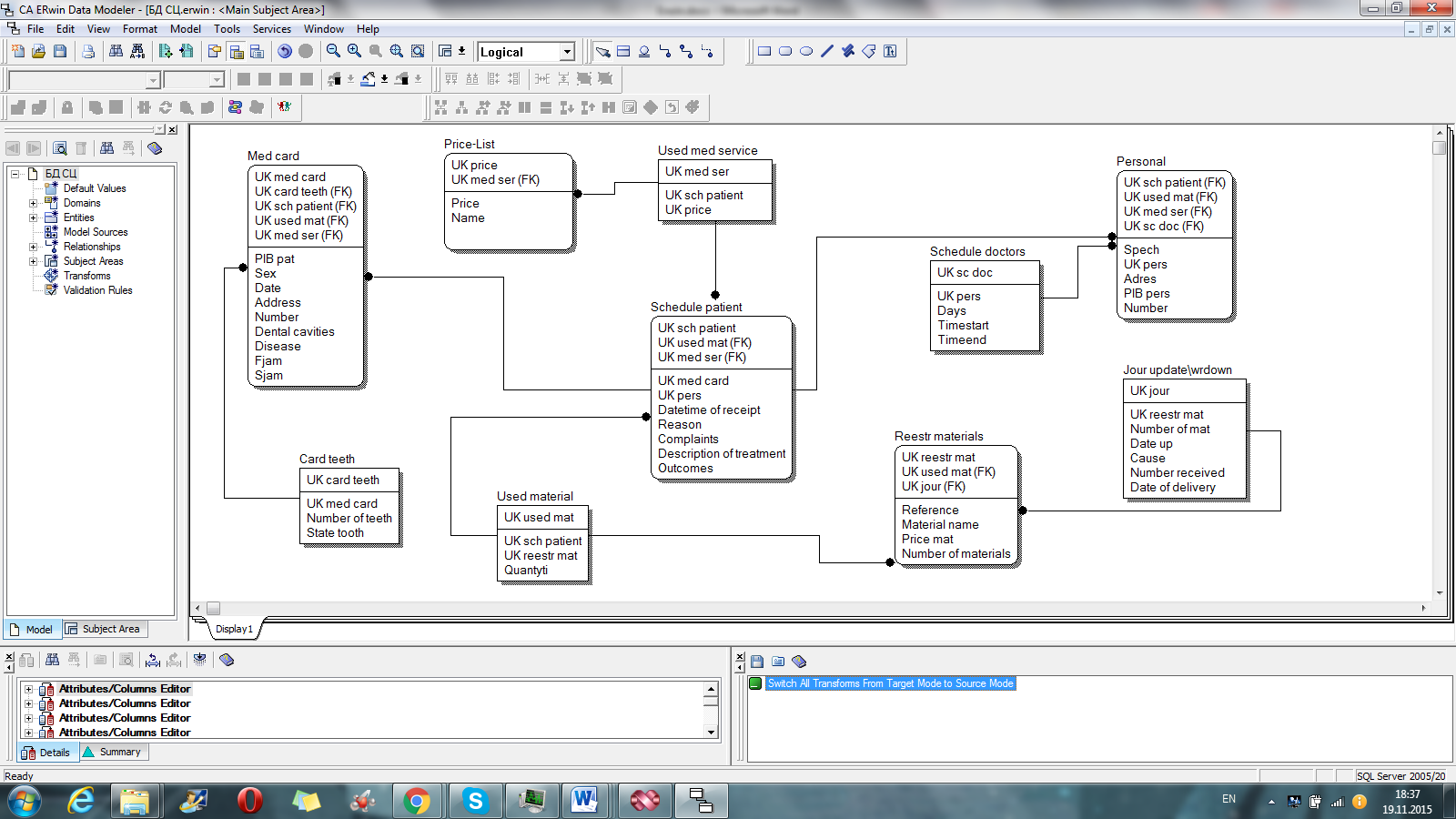


Рисунок. 2.2.1 - Концептуальна модель

Сукупність < Реєстр матеріалу > представляє собою реєстр матеріалів, які були використані. Ключовим атрибутом , який дає доступ до об'єкта <: Реєстр матеріалу >, є унікальний атрибут <УК реєстр матеріалу >. Сукупність < Реєстр матеріалу > асоційований з сукупністю < Використаний матеріал > за допомогою атрибута < УК реєстр матеріалу > ; та з сукупністю <Журнал онов./спис. > за допомогою атрибута < УК реєстр матеріалу >.

Сукупність < Журнал онов./спис. > представляє собою журнал оновлення/списання матеріалів. Ключовим атрибутом , який дає доступ до об'єкта <: Журнал онов./спис. >, є унікальний атрибут <УК журнал онов./спис.>. Сукупність < Журнал онов./спис.> асоційований з сукупністю < Реєстр матеріалу > за допомогою атрибута < УК реєстр матеріалу >.

Сукупність < Зубна карта > представляє собою зубну карту пацієнта . Ключовим атрибутом , який дає доступ до об'єкта <: Зубна карта >, є унікальний атрибут <УК зубна карта >. Сукупність < Зубна карта > асоційований з сукупністю < Медична карта пацієнта > за допомогою атрибута < УК медична карта пацієнта >.

### 2.2.2 Даталогічна модель АІС

Даталогічна модель даних описує реалізацію об'єктів логічної моделі на рівні об'єктів конкретної бази даних.

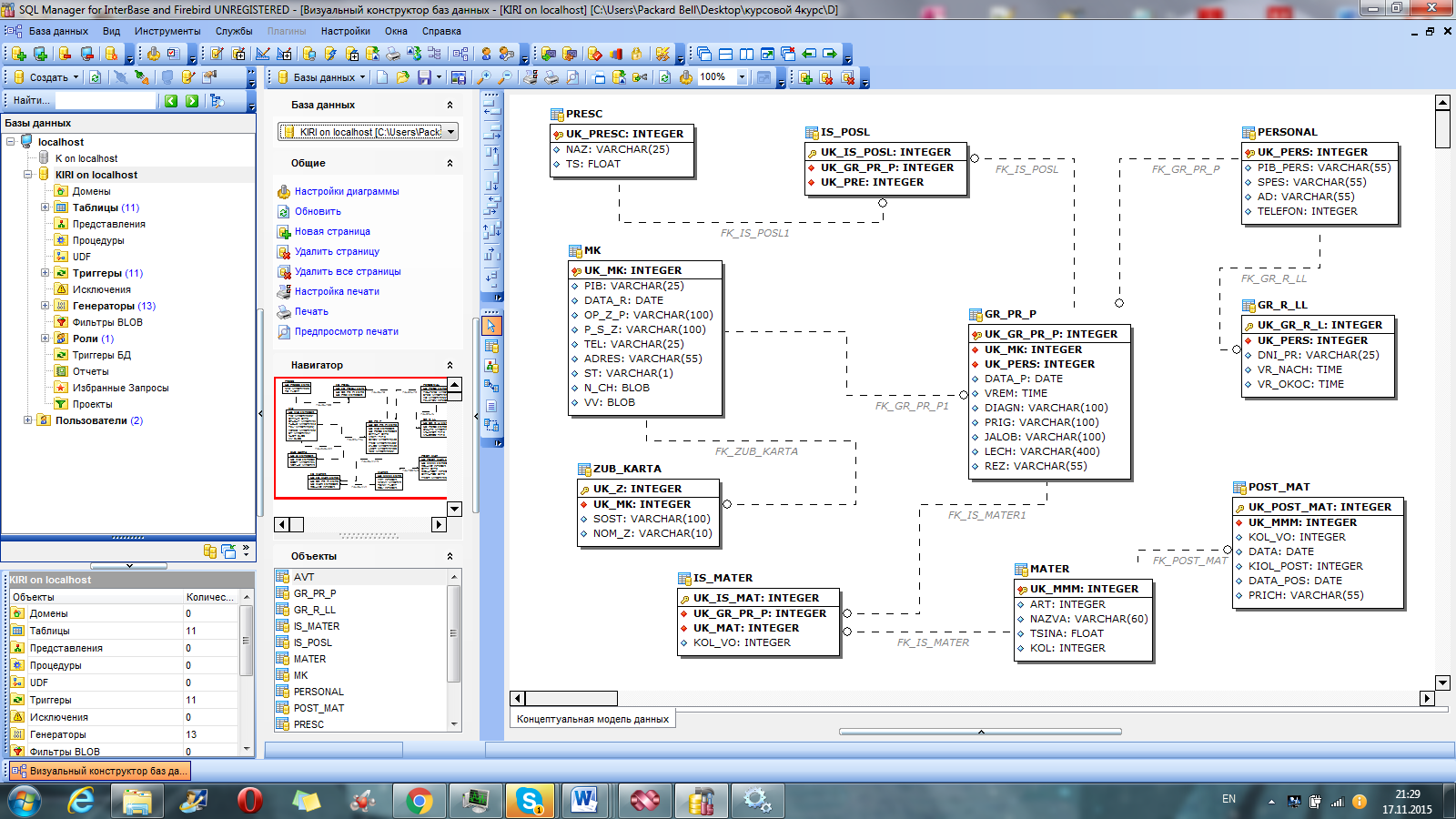


Рисунок 2.2.2 – Даталогічна модель БД

## 2.3 Математичне забезпечення

### 2.3.1 Розробка алгоритмів рішення функціональної задачі

Поведінкова модель системи за методологією Flow Chart

Схеми алгоритмів, програм, даних і систем (далі - схеми) складатися з заданого значення символів, короткого пояснювального тексту і з'єднуючих ліній.

Схеми можуть використовуватися на різних рівнях деталізації, причому число рівнів залежить від розмірів і складності завдання обробки даних. Рівень деталізації повинен бути таким, щоб різні частини і взаємозв'язок між ними були зрозумілі в цілому.

Графічні символи, призначені для використання в документації з обробки даних, застосовуються в:

- схемах даних;

- схемах програм;

- схемах роботи системи;

- схемах взаємодії програм;

- схемах ресурсів системи.

Опис схем

Схеми даних відображають шлях даних при вирішенні завдань і визначають етапи обробки, а також різні приємним носії даних.

Схема даних складається з:

* символів даних (символи даних можуть також вказувати вид даних);
* символів процесу, який слід виконати над даними (символи
* процесу можуть також вказувати функції, виконувані

обчислюваною машиною);

* символів ліній, що вказують потоки даних між процесами і (або)

носіями даних;

* спеціальних символів, що використовуються для полегшення

написання і читання схеми.

Символи даних передують і слідують за символами процесу. Схема даних починається і закінчується символами даних (за виключенням спеціальних символів) (рис.2.3.2.3).

Схеми програм відображають послідовність операцій в про-грамі.

Схема програми складається з:

- символів процесу, що вказують фактичні операції обробки даних (включаючи символи, що визначають шлях, якого слід дотримуватися з урахуванням логічних умов);

- лінійних символів, що вказують потік управління;

- спеціальних символів, що використовуються для полегшення написання і читання схеми. (рис.2.3.2.4)

Схеми роботи системи відображають управління операціями і потік даних у системі.

Схема роботи системи складається з:

- символів даних, що вказують на наявність даних (символи даних можуть також вказувати вид носія даних);

- символів процесу, що вказують операції, які слід виконувати над даними, а також визначають логічний шлях, якого слід дотримуватися;

- лінійних символів, що вказують потоки даних між процесами і (або) носіями даних, а також потік управління між процес-самі;

- спеціальних символів, що використовуються для полегшення написання і читання блок-схеми. (рис.2.3.2.5)

Схеми взаємодії програм відображають шлях активації про-грам і взаємодій з відповідними даними. Кожна програма в схемі взаємодії програм показується тільки один раз (у схемі роботи системи програма може зображуватися більш ніж в одному потоці управління).

Схема взаємодії програм складається з:

- символів даних, що вказують на наявність даних;

- символів процесу, що вказують на операції, які слід ви-повнити над даними;

- лінійних символів, що відображають потік між процесами та даними, а також ініціації процесів;

- спеціальних символів, що використовуються для полегшення написання і читання схеми. (рис.2.3.2.6)

Схема ресурсів системи відображають конфігурацію блоків даних і обробних блоків, яка потрібна для вирішення завдання або набору завдань.

Схема ресурсів системи складається з:

- символів даних, що відображають вхідні, вихідні та запам'ятовуючі пристрої обчислювальної машини;

- символів процесу, що відображають процесори (центральні процесори, канали);

- лінійних символів, що відображають передачу даних між вводу-виводу і процесорами, а також передачу управління між процесорами;

- спеціальних символів, що використовуються для полегшення написання і читання схеми. (рис.2.3.2.7)

Розглядаємо бізнес процеси «Огляд матеріалів», «Огляд пацієнта».

Розпочнемо з блок – схеми «Огляд пацієнта» з раніше розглядоваємої діаграми IDEF0.

Після терміналу проходить процес «Опитування пацієнта» після неї наступний процес «Початок огляду». Далі потік даних переходить до рішення « Усі необхідні матеріали є?» і в залежності від наявності матеріалів потік даних розподіляється на «Ні» - йде процес «Заказ необхідних матеріалів», інформація береться з БД огляд матеріалів, а далі процес «Формування списку необхідних матеріалів» і потік даних знову йде на початок рішення « Усі необхідні матеріали є?» .

У випадку потоку даних «Так» процес «Підготовка інструментів», далі - «Огляд пацієнта» , потік даних переходить до процесу «Огляд порожнини рота». Після цього в залежності від результату огляду порожнини рота, потік даних розподіляється на процес «Виконання профілактичних засобів» та йде до кінцевого терміналу.

А інший потік даних на рішення «Лікування?», далі також розподіляється на два потоки, на процес «Лікування» - до кінцевого терміналу. А інша на процес «Закінчення огляду». після цього уся інформація вводиться в медичну карту, та взаємодіє з базою даних. Після цього йде кінцевий термінал. (рис. 2.3.2 .1).

Далі блок –схема «Огляд обліку матеріалів» з раніше розглядоваємої діаграми DFD.

Після початкового терміналу потік операцій переходить до процесу «Огляд усіх матеріалів» - БД обліку матеріалів. Далі – рішення і потік даних «Так» - заказ необхідних матеріалів» -«Оформлення заказу постачальників» -БД огляд матеріалів на кінцевий термінал. Усі потоки даних, які відгалужуються від основного після обробки закінчуються кінцевим терміналом і розглядаються як окремі процеси, не зв`язані повністю з основним потоком даних та операцій. Після перевірки журналу оновлень та списань - БД журналу списань та оновлень, дані або йдуть на оформлення списків та на кінцевий термінал чи на перевірку списків необхідних матеріалів –БД списку необхідних матеріалів, потік даних переходить до процесу «Видача...» та закінчується кінцевим терміналом. (рис 2.3.2.2).



Рис. 2.3.2.1 – Схема бізнес- процесу «Огляд матеріалів»



Рис.2.3.2.2 – Схема бізнес – процесу «Огляд пацієнта»



Рис. 2.3.2.3 – Схема даних



Рис. 2.3.2.4 Схема програми



Рис. 2.3.2.5 Схема роботи системи



Рис. 2.3.2.6 - Схема взаємодії системи



Рис. 2.3.2.7 – Схема ресурсів

## 2.4 Програмне забезпечення

Обґрунтування вибору СКБД

Для розроблюваного мною завдання використовується клієнт-серверна архітектура, тому що у системі будуть працювати багато користувачів. А завдання або мережева навантаження будуть розподілені між постачальниками послуг, званими серверами, і замовниками послуг, званими клієнтами.

Так як архітектура програмного забезпечення для поставленого завдання буде клієнт-серверною, то в якості СКБД, мною була обрана Firebird.

Технології, на яких заснована Firebird використовуються більше 20 років, що зробило її вельми зрілим і стабільним продуктом.

Firebird - це вільна система керування базами даних (СКБД). Це проект, незалежний, з комерційної точки зору. Заснована Firebird на вихідному коді вільної версії СКБД Interbase 6.0, виданої компанією Borland 25 липня 2000. Працює Firebird на Microsoft Windows, Linux і всіляких Unix платформах.

Firebird - це повнофункціональна і потужна СКБД, вона може обслуговувати бази даних розміром від кількох кілобайт до багатьох гігабайт, показуючи хорошу продуктивність і практично не потребуючи обслуговування.

Багатоверсійна архітектура, яка є безперечною перевагою Firebird, забезпечує паралельну обробку аналітичних і оперативних запитів (справа в тому, що читають і пишуть користувачі не блокують один одного). Плюсами Firebird є компактність, висока ефективність і потужна мовна підтримка для збережених тригерів і процедур. Firebird легко підтримує досить великі бази даних. Серед мінусів найчастіше називають відсутність кеша підсумків запитів, повнотекстових індексів.

Система керування базами даних Firebird використовується в самих різних промислових системах як державного , так і недержавного сектора.

Варто підкреслити, що Firebird - це сервер обробки баз даних, і один такий сервер може обробити сотні незалежних баз даних, і кожну - з величезною кількістю користувальницьких з'єднань. Щоб забезпечити резервне копіювання, сервер зупиняти немає необхідності. Сам процес резервного копіювання зберігає стан бази даних на момент старту, при цьому ніяк не перешкоджаючи роботі з базою.

Firebird підтримує більшість міжнародних наборів символів з ​​величезною кількістю варіантів сортування.

Зважаючи на всі переваги та особливості Firebird, я вважаю вибір цієї СКБД доцільним.

Обґрунтування вибору мови програмування.

В якості мови програмування для реалізації поставленого завдання була обрана C++, оскільки вона володіє наступними відмінними характеристиками.

C++ була розвинута з мови програмування C і за дуже малими виключеннями зберігає C як підмножину. Базова мова, C підмножина C++, спроектована так, що існує дуже близька відповідність між його типами, операціями й операторами і комп'ютерними об'єктами, з якими безпосередньо приходиться мати справу: числами, символами й адресами.

Це універсальна мова програмування високого рівня з підтримкою декількох парадигм програмування: об'єктно-орієнтованої, узагальненої та процедурної. Розроблена Б'ярном Страуструпом в AT&T Bell Laboratories у 1979 році. Базується на мові С. У 1990-х роках С++ стала однією з найуживаніших мов програмування загального призначення.

С++ є мовою програмування загального призначення. Природна для неї область застосування - системне програмування, що розуміється в широкому сенсі цього слова. Крім того, С + + успішно використовується в багатьох областях додатків, що далеко виходять за зазначені рамки. Реалізація С++ тепер є на всіх машинах, починаючи з самих скромних мікрокомп'ютерів - до найбільших супер - ЕОМ і практично для всіх операційних систем.

Мова С++ проектувалася для підтримки абстракції даних і об'єктно -орієнтованого програмування в додавання до традиційного стилю С.

За винятком другорядних деталей вона містить мову С як підмножина. Мова С розширюється введенням гнучких і ефективних засобів, призначених для побудови нових типів. Програміст структурує своє завдання, визначивши нові типи, які точно відповідають поняттям предметної області завдання. Такий метод побудови програми зазвичай називають абстракцією даних. Інформація про типи міститься в деяких об'єктах типів, визначених користувачем. З такими об'єктами можна працювати надійно і просто навіть у тих випадках, коли їх тип не можна встановити на стадії трансляції. Програмування з використанням таких об'єктів зазвичай називають об'єктно - орієнтованим. Якщо цей метод застосовується правильно, то програми стають коротшими і зрозуміліше, а супровід їх спрощується.

    Ключовим поняттям С++ клас. Клас - це визначений користувачем тип. Класи забезпечують упорядкування даних, їх ініціалізацію, неявне перетворення користувачем типів, динамічне завдання типів, контрольоване користувачем керування пам'яттю і засоби для перевантаження операцій. У мові С++ концепції контролю типів і модульної побудови програм реалізовані більш повно, ніж у С.

Крім того, С++ містить удосконалення, прямо з класами не зв'язані: символічні константи, функції - підстановки, стандартні значення параметрів функцій, перевантаження імен функцій, операції керування вільною пам'яттю і контрольний тип. У С++ збережені всі можливості з ефективної роботи з основними об'єктами, що відображають апаратну "реальність" (розряди, байти, слова, адреси). Це дозволяє досить ефективно реалізовувати типи.

   С++ проектувалася для підтримки того принципу, що всяка програма є модель деяких існуючих в реальності понять, а клас є конкретним поданням поняття, взятого з області додатків. Тому класи пронизують всю програму на С++, і накладаються жорсткі вимоги на гнучкість поняття класу, компактність об'єктів класу та ефективність їх використання. Якщо працювати з класами буде незручно або занадто накладно, то вони просто не будуть використовуватися, і програми виродяться в програми на "кращому С". Значить користувач не зуміє насолодитися тими можливостями, заради яких, власне, і створювалася мова.

 У мові C++ повністю підтримуються принципи об'єктно -орієнтованого програмування, включаючи три основи, на яких воно складається: інкапсуляцію, спадкування і поліморфізм.

Перевагами мови C++ є:

* швидкодія. Швидкість роботи програм на С++ практично не

поступається програмам на С, хоча програмісти отримали в свої руки нові можливості і нові засоби;

* масштабованість. На мові C++ розробляють програми для

найрізноманітніших платформ і систем;

* можливість роботи на низькому рівні з пам'яттю, адресами, портами;
* можливість створення узагальнених алгоритмів для різних типів даних,

їхня спеціалізація, і обчислення на етапі компіляції, з використанням шаблонів;

* підтримуються різні стилі та технології програмування, включаючи

традиційне директивне програмування, ООП, узагальнене програмування, метапрограмування (шаблони, макроси).

Мова C++ стала потужним і стрімким ривком у розвитку програмування. C++ і донині займає панівне положення серед мов програмування в світі. Тому я буду використовувати саме цю мову при розробці проекту

Обґрунтування вибору інструментальних засобів програмної реалізації.

Інструментальним засобом програмної реалізації, я обрав Embarcadero RAD Studio, тому що вона являє собою повнофункціональний набір засобів розробки, який дозволяє швидко у візуальному режимі створювати додатки з розвиненим інтерфейсом для Windows, NET, PHP та веб - рішень. RAD Studio включає в себе Delphi, C++ Builder; підтримує компільовані, керовані і динамічні мови, підключення до баз даних в різнорідної середовищі, повнофункціональні платформи візуальної розробки, а також обширну екосистему ПЗ сторонніх розробників. Все це дає можливість значно прискорити розробку додатків для різних платформ Windows, веб - рішень і баз даних.

Середовища розробки RAD Studio багаторазово спрощують і прискорюють створення візуальних додатків і додатків, що активно працюють з базами даних, - від додатків з розвиненим інтерфейсом для настільних систем і сенсорних екранів до хмарних рішень, керованих базами даних багатоланкових систем, веб -додатків і служб.

RAD Studio - це єдиний набір засобів для швидкої розробки додатків Windows, NET і веб - додатків на PHP. Він об'єднує в своєму складі три інтегрованих середовища розробки, що забезпечує оптимальні і найбільш ефективні умови для роботи з будь-якою платформою. Вбудовані засоби аналізу допомагають підтримувати високу продуктивність додатків на етапі виконання. Компілюючи в машинний код, програми Delphi і C++ Builder володіють достатньою швидкодією і обчислювальною потужністю для візуалізації даних, керування апаратним забезпеченням в режимі реального часу і обробки тривимірних об'єктів.

RAD Studio надає в розпорядження розробника широкий набір компонентів. Сотні вбудованих компонентів з бібліотеки VCL, а також тисячі безкоштовних і комерційних сторонніх компонентів дозволяють створювати в Delphi і C++ Builder додатки практично будь-якого типу і призначення, значно інтенсифікуючи процес розробки.

Embarcadero включила до складу пакету більше 500 вільно розповсюджуваних і платних шаблонних компонентів з бібліотеки VCL, а також кількох інших джерел, які підходять для створення практично будь-якої програми. У складі RadPHP входить єдина на ринку бібліотека візуальних компонентів для розробки на PHP, що дозволяє використовувати з цією мовою веб - програмування знайомий багатьом по простоті інтерфейс середовища Delphi. Модулі збірки, що відповідають за роботу з Delphi і C++ - додатками, розраховані на взаємодію з СКБД, багаторівневої архітектурою і « хмарними » платформами.

 Варто також відзначити, що в комплект поставки RAD Studio було включено систему керування базами даних Firebird, в якій є відкритий вихідний код. Для деяких промислових СКБД були розроблені оновлені драйвери.

Основними перевагами є:

* швидший спосіб створення високопродуктивних додатків для різних

платформ;

* скорочення термінів розробки за допомогою вбудованих компонентів;
* значне скорочення термінів розробки. RAD Studio дозволяє створювати

додатки для Windows в п'ять разів швидше, ніж інші рішення для розробки, або витрачати на це в п'ять разів менше ресурсів. Застосування RAD Studio зменшує витрати часу і ресурсів на створення додатків, оскільки всі функції середовища розробки RAD Studio підпорядковані одній меті: прискорити створення програм. А середовище швидкої розробки дозволяє зменшити обсяг коду, необхідного для вирішення завдань, що стоять перед розробниками, забезпечуючи досягнення високих результатів при менших витратах.

* підвищення ефективності завдяки повторному використанню коду. Код

і компоненти в RAD Studio спочатку спроектовані для повторного використання.

* підтримка різних мов.
* скорочення шляху від прототипу до готової версії. Швидка розробка

прототипів прискорює практичну реалізацію ідей і дозволяє з мінімальними трудовитратами втілювати ідеї в готові версії продукту.

* підтримка більшої кількості користувальницьких настільних систем.

Інфраструктура VCL підтримує різні версії Windows, дозволяючи підтримувати більше настільних систем Windows, не піклуючись про особливості реалізації Windows API кожної версії.

Для мене Embarcadero RAD Studio принесла набір актуальних і затребуваних інструментів. Цікавим виглядає і інтеграція продукту з «хмарними» рішеннями, а також власний елемент адміністрування СКБД.

### 2.4.1 Схема взаємодії програмних модулів

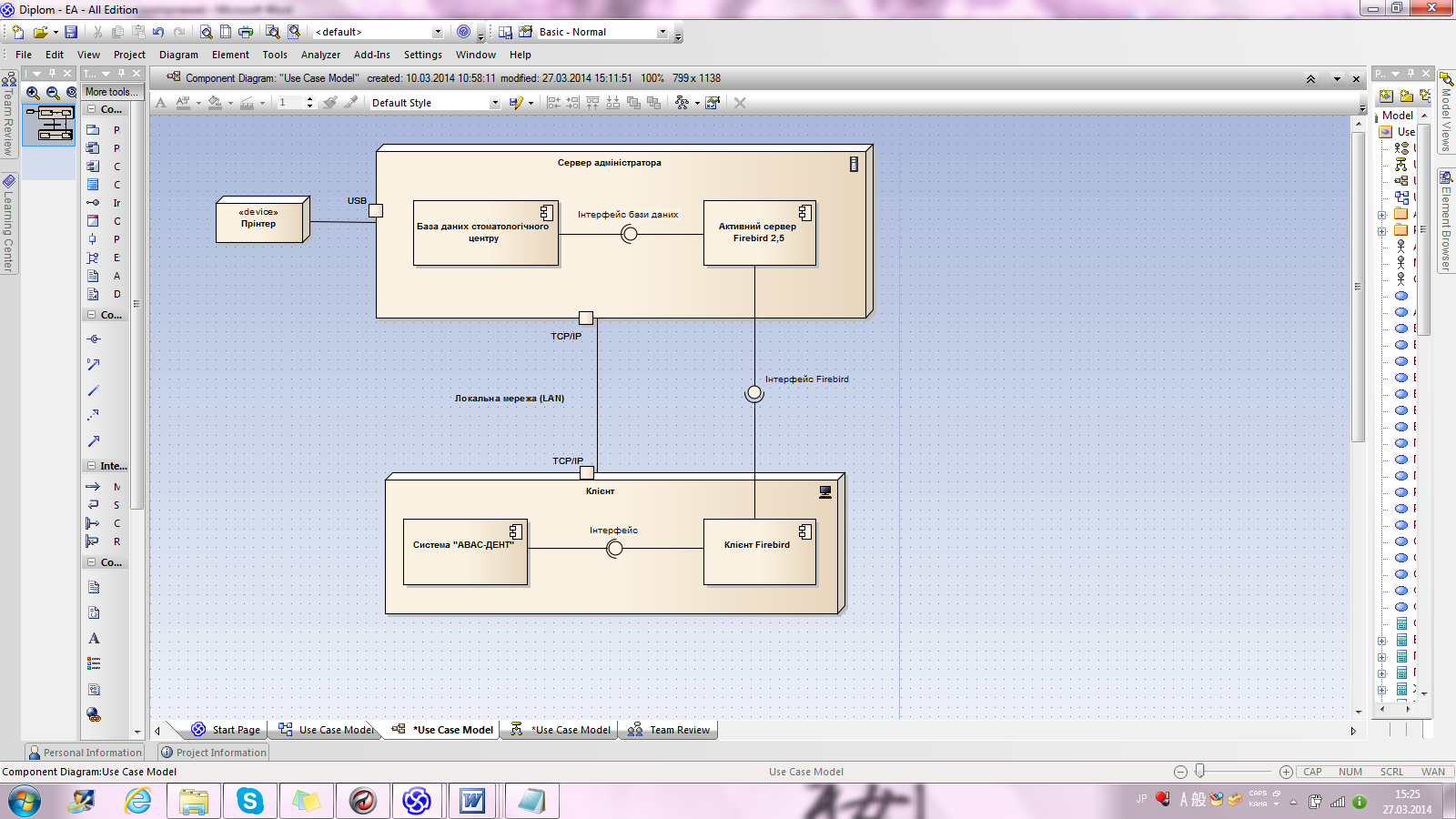


Рисунок 24.1 – Схема взаємодії програмних модулів

### 2.4.2 Архітектура системи

В основі проектування АС лежить клієнт-серверна архітектура.

Клієнт-сервер являє собою обчислювальну або мережеву архітектуру, в якій завдання або мережева навантаження розподілені між постачальниками послуг, які називаються серверами, і замовниками послуг, званих клієнтами. Нерідко клієнти і сервери взаємодіють через комп'ютерну мережу і можуть бути як різними фізичними пристроями, так і програмним забезпеченням.

Дана архітектура має ряд переваг:

* Робить можливим, в більшості випадків, розподілити функції обчислювальної системи між декількома незалежними комп'ютерами в мережі. Це дозволяє спростити обслуговування обчислювальної системи. Зокрема, заміна, ремонт, модернізація або переміщення сервера, не зачіпають клієнтів.
* Всі дані зберігаються на сервері, який, як правило, захищений набагато краще за більшість клієнтів. На сервері простіше забезпечити контроль повноважень, щоб дозволити доступ до даних тільки клієнтам з відповідними правами доступу.
* Дозволяє об'єднати різні клієнти. Використовувати ресурси одного сервера часто можуть клієнти з різними апаратними платформами, операційними системами.

Поряд з достоїнствами, існують і недоліки архітектури:

* Непрацездатність сервера може зробити непрацездатною всю обчислювальну мережу.
* Підтримка роботи даної системи вимагає окремого спеціаліста - системного адміністратора.
* Висока вартість обладнання.

#### 2.4.2.1 Діаграма компонентів

Діаграма компонентів та розгортань зроблена разом та сформована в одну діаграму.

#### 2.4.2.2 Діаграма розгортання

Елементи логічного представлення (Класи та асоціації) не існує фізично, а лише відображають розуміння структури системи або аспекти її поведінки. У UML для цих цілей є два види діаграм: діаграма розгортання та діаграма компонентів.

Хоча діаграми розгортання та діаграми компонентів можна зображати окремо, також допускається поміщати діаграму компонентів на діаграму розгортання. Це доцільно робити , щоб показати які компоненти виконуються і на яких вузлах.

Діаграма компонентів показує залежності і взаємодії між компонентами програмного забезпечення. Вона зображує фізичну архітектуру комп'ютера базової системи. Діаграма компонентів розробляється для візуалізації загальної структури вихідного програмного коду і специфікації збірки виконуваного програмного коду системи. Основними графічними елементами діаграми компонентів є компоненти, інтерфейси і залежності. Діаграма компонентів дозволяє визначити архітектуру системи, що розробляється , встановивши залежності між програмними компонентами.

Компонент - це контейнер пакетів, що беруть участь в роботі системи. Робочі компоненти представлені пакетами вихідного і виконуваного коду. Для взаємодії компонентів системи компоненти мають порти (для організації взаємодії розподілених компонентів) і реалізують деякий набір інтерфейсів.

Діаграма компонентів (розгортань) моделі проектованої системи виглядає, як це показано на рисунку 2.4.2.2.

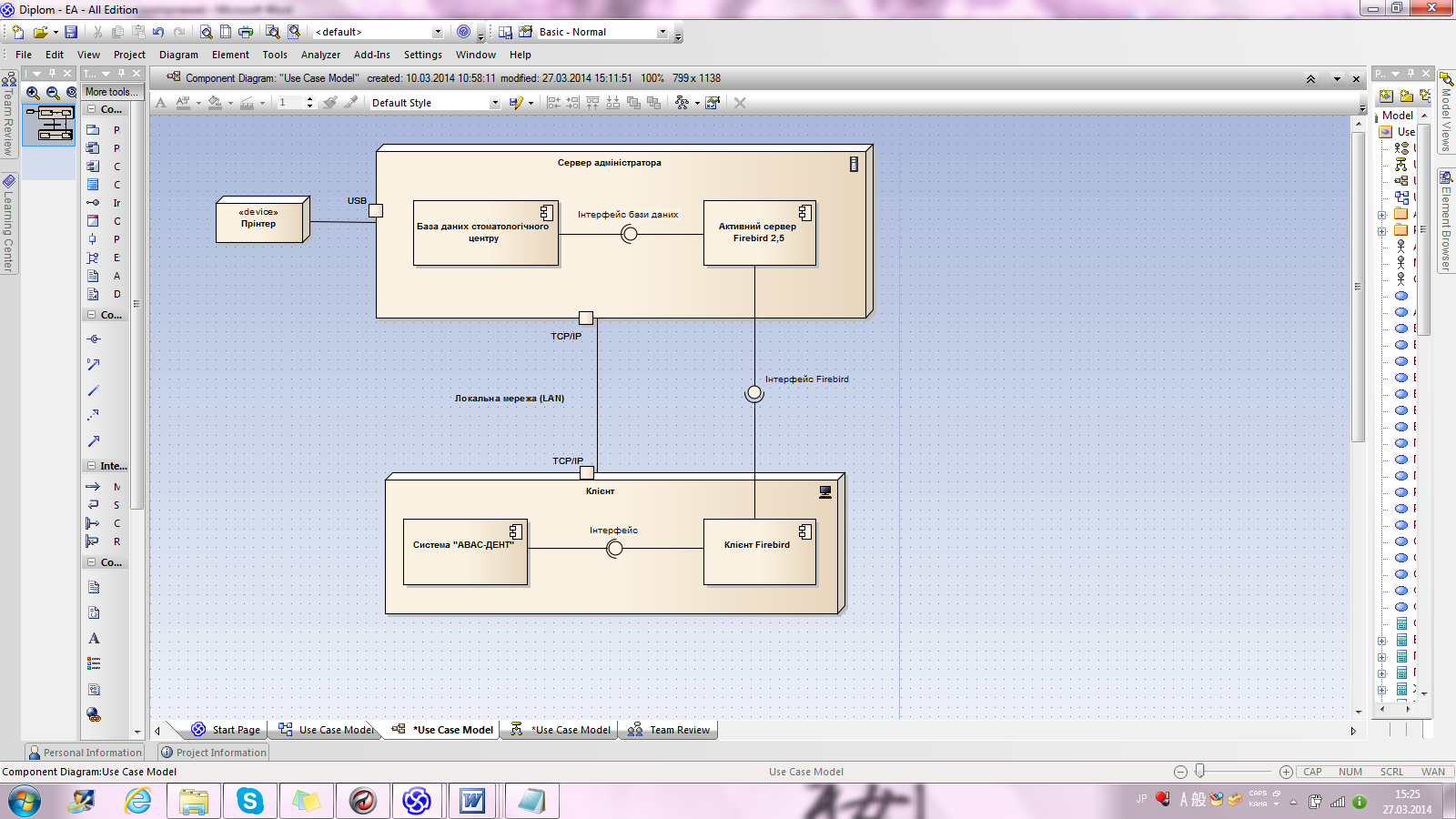


Рисунок 2.4.2.2 – Діаграма компонентів (розгортань) системи

Діаграми розгортання - це один з двох видів діаграм, що використовуються при моделюванні фізичних аспектів об'єктно - орієнтованої системи. Така діаграма показує конфігурацію вузлів, де проводиться обробка інформації , і те, які компоненти розміщені на кожному вузлі.

Діаграми розгортання використовуються для моделювання статичного вигляду системи з точки зору розгортання. В основному під цим розуміється моделювання топології апаратних засобів , на яких виконується система.

Діаграми розгортання важливі не тільки для візуалізації, специфікації та документування вбудованих, клієнт - серверних і розподілених систем, але і для керування виконавчими системами з використанням прямого і зворотного проектування.

## 2.5 Організаційне забезпечення

### 2.5.1 Інструкція користувача

Програма АІС стоматологічного «АВАС-ДЕНТ» призначена для введення обліку роботи персоналу та обліку матеріалів стоматологічного центру.

Запуск програми в операційній системі сімейства Windows здійснюється одним з стандартних способів:

* подвійним клацанням лівою кнопкою миші на ярлику програми;
* викликом контекстного меню з вибором його пункту «Открыть».

Після запуску програми на екрані монітора з'являється форма авторизації. На рисунку 2.5.1 зображено вікно форми авторизації.

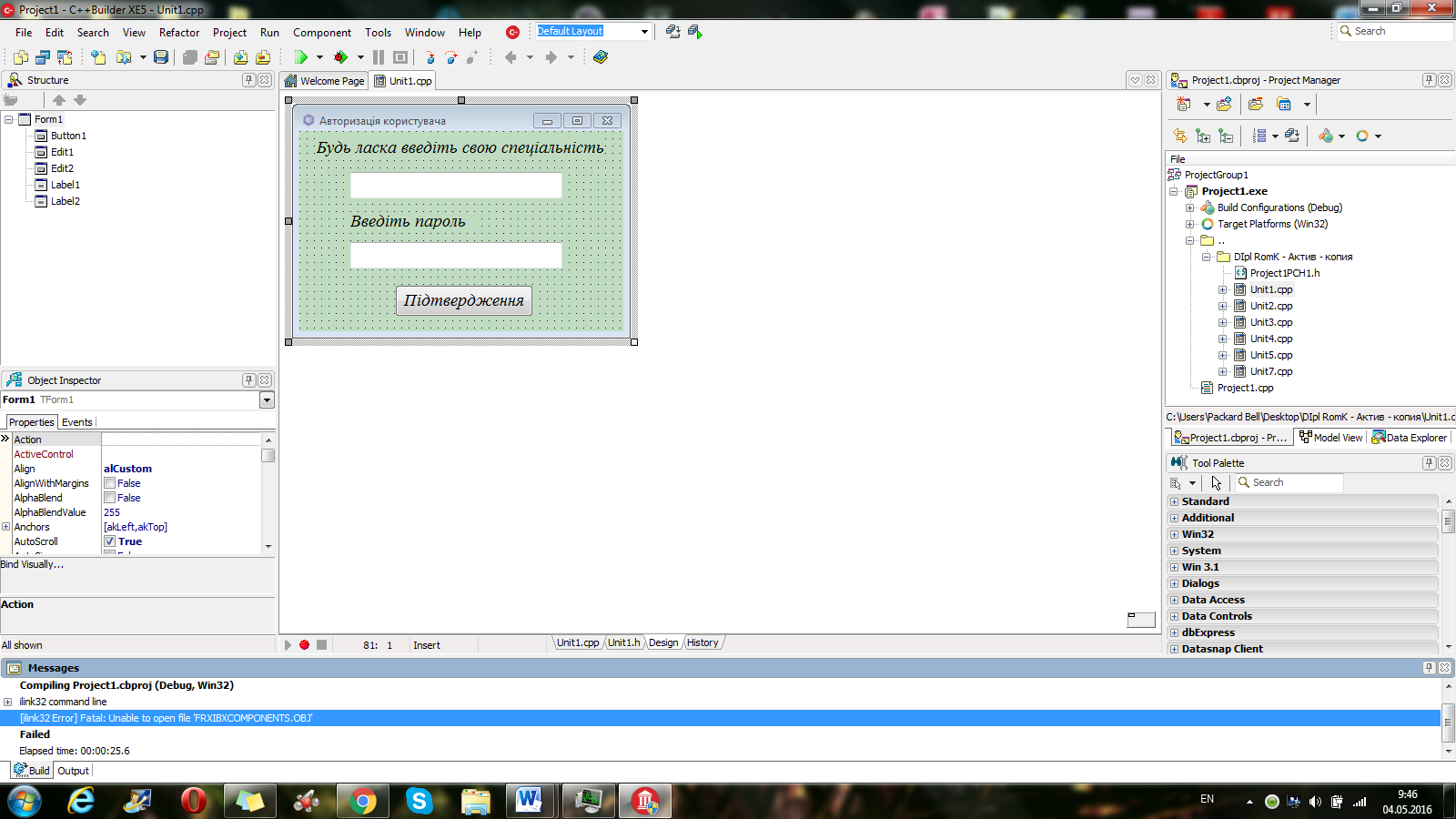


Рисунок 2.5.1 – Вікно «Авторизація користувача»

У програмі передбачена авторизація користувачів. Модулі системи повинні доступні користувачам у відповідності з рівнем доступу користувача до системи. Після підтвердження авторизації, буде відкрито вікно «Головна

форма «АВАС-ДЕНТ»». На рисунку 2.5.1.1 зображено вікно Головна форма «АВАС-ДЕНТ».

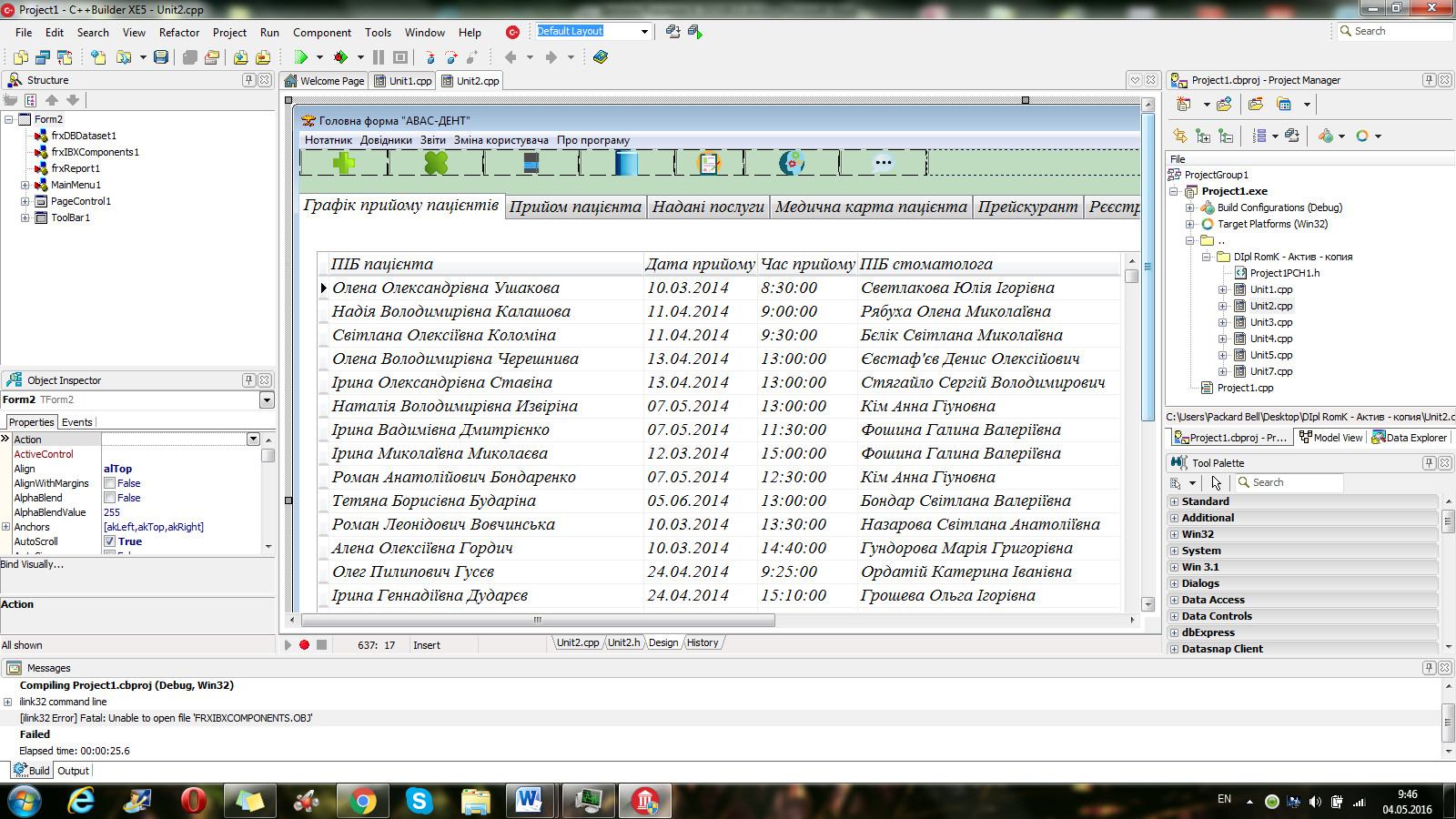


Рисунок 2.5.1.2– Вікно «Головна форма «АВАС-ДЕНТ»»

Графічний інтерфейс програми складається з головного вікна та додаткових вікон. Головне вікно програми має панель меню, кнопки та вкладки. Меню програми містить усі команди для керування її виконанням.

Складається з таких вкладок: «Графік прийому пацієнтів», «Медична карта пацієнта», «Прейскурант», «Реєстр матеріалів», «Журнал онов./спис.», «Персонал». Для всіх вкладок вікна «Головна форма АВАС-ДЕНТ» будуть доступні кнопки: «пошук», «фільтрація», «звіти», «вихід».

При натисканні кнопки «Фільтрація» буде відбуватися фільтрація за введеними даними.

При натисканні кнопки «Відміна фільтрації» буде відбуватися відміна фільтрації за введеними даними.

Кнопка «Пошук» буде викликати вікно пошуку, завдяки якому буде відбуватися пошук за введеними даними.

Кнопка «Звіти» буде давати можливість адміністратору переглянути та надрукувати чек про оплату послуг. На рисунку 2.5.1.2 зображено вікно звіту «Прейскурант».

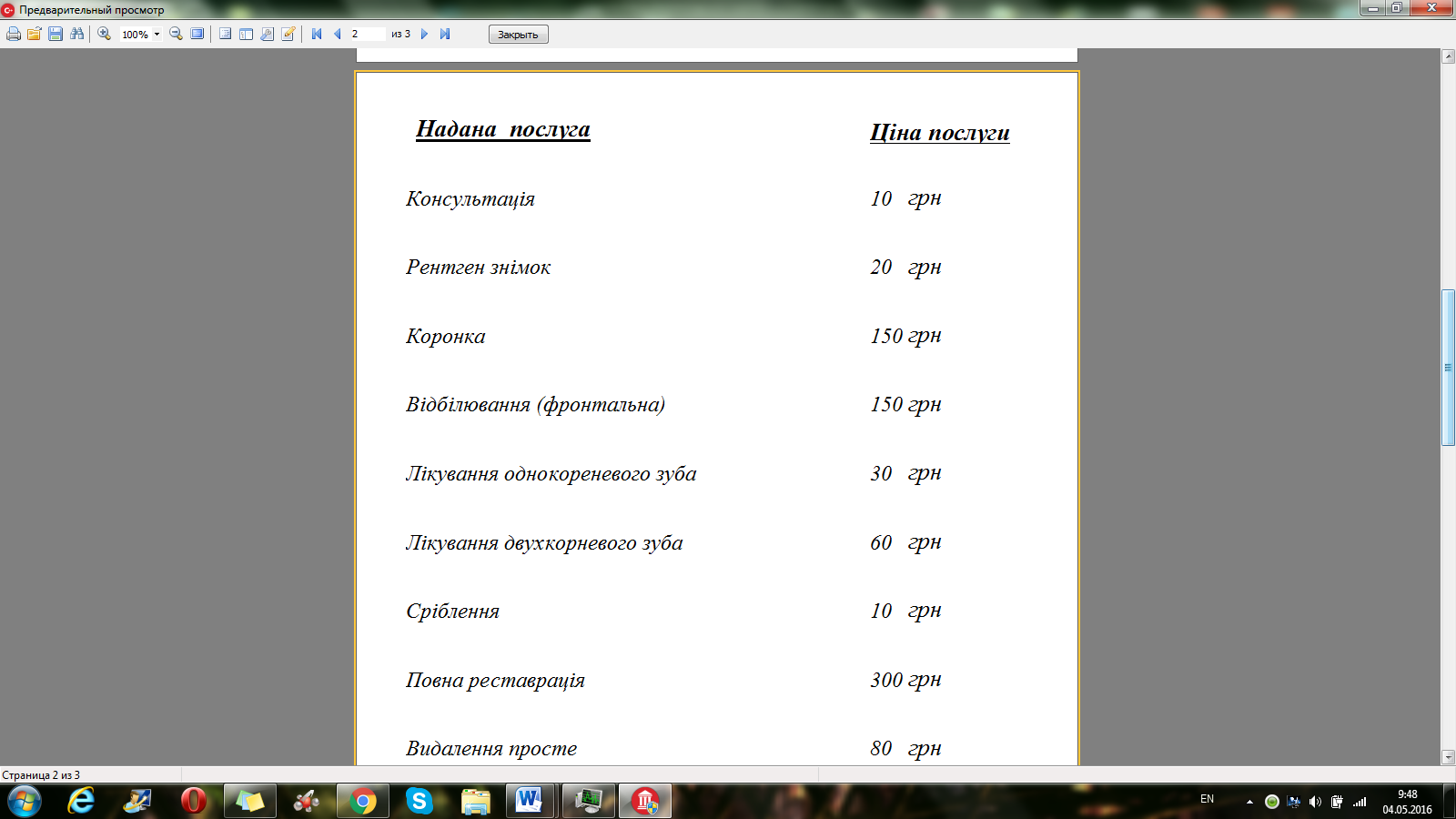


Рисунок 2.5.1.3 – Вікно звіту «Чек про оплату»

При натисканні кнопки «Вихід» буде організовано вихід з програми.

При натисканні кнопки «Про програму» можна буде переглянути вікно «Про програму».

### 2.5.2 Інструкція адміністратора

Для установки автоматизованої системи обліку роботи стоматологічного центру поліклініки необхідно відкрити інсталяційний диск, на якому записані 2 файли: Install\_Avas-Dent.exe і папка «для Firebird».

На комп'ютер, який встановлюється система, повинна бути встановлена ​​СУБД Firebird, яка є безкоштовною (за винятком деяких комерційних версій). Після запуску файлу Install\_Avas-Dent.exe починається установка програми в обрану адміністратором папку . Після вибору директорії для установки відкривається вікно, що містить інформацію про відповідальність за порушення авторських прав . У ході установки будуть створені і відредаговані необхідні таблиці в базі даних, встановлені всі драйвера і настройки за замовчуванням. Після завершення установки необхідно перезавантажити комп'ютер. Адміністратор призначає логін і пароль для кожного із співробітників. Також у адміністратора є можливість контролювати усю роботу системи.

## 2.6 Технічне забезпечення

### 2.6.1 Обґрунтування вибору (розробки) технічного забезпечення АІС

Надійне (стійке) функціонування програми має бути забезпечене виконанням користувачем сукупності організаційно - технічних заходів, перелік яких наведено нижче:

* організацією безперебійного живлення технічних засобів;
* використанням ліцензійного програмного забезпечення;
* регулярним виконанням рекомендацій про типові норми часу на роботи по сервісному обслуговуванню ПЕОМ та оргтехніки і супроводу програмних засобів;
* регулярним виконанням вимог щодо захисту інформації та випробуваннях програмних засобів на наявність комп'ютерних вірусів.

Час відновлення після відмови, викликаного збієм електроживлення технічних засобів (іншими зовнішніми факторами), що не є фатальним збієм (не крах) операційної системи , не повинно перевищувати 30 -ти хвилин за умови дотримання умов експлуатації технічних і програмних засобів. Час відновлення після відмови, викликаного несправністю технічних засобів, фатальним збієм (крахом) операційної системи, не повинно перевищувати часу, необхідного на усунення несправностей технічних засобів і переустановлення програмних засобів.

Відмови програми можливі внаслідок некоректних дій оператора (користувача) при взаємодії з операційною системою. Щоб уникнути виникнення відмов програми за вказаною вище причини слід забезпечити роботу кінцевого користувача без надання йому адміністративних привілеїв.

З розроблюваної системою повинно мати можливість працювати десять користувачів , тому повинна бути обрана клієнт-серверна технологія зберігання та обробки інформації.

СКБД повинна забезпечувати можливість роботи у багатокористувацькому режимі, відновлення працездатності при програмно - апаратних збіях, таких як відключення електроживлення, проблеми з апаратним забезпеченням, нештатне завершення роботи .

З метою забезпечення надійного функціонування в СКБД має бути передбачено :

* збереження працездатності системи при некоректних операціях користувача (введення некоректних даних);
* збереження цілісності даних при позаштатному завершенні роботи системи;
* резервне копіювання даних;
* журналювання операційної системи.

Даним вимогам відповідають такі СКБД, як, SQL Server, Oracle, InterBase, Firebird. Вибір інших СКБД недоцільний.[3]

Вимоги до сервера: процесор Intel Core, мінімальний обсяг оперативної пам'яті 2 Гб, мінімальний обсяг дискового простору 60 ГБ, операційна система Windows XP/Vista/7.

Вимоги до клієнтської машині: процесор Intel Core, мінімальний обсяг оперативної пам'яті 2 ГБ, мінімальний обсяг дискового простору 60 ГБ, операційна система Windows XP/Vista/7. Та периферійний пристрій – принтер.

Розробляється система повинна передбачати можливість її подальшого розвитку, модифікації і включення нових функцій в систему, поліпшення коду, можливість розширення механізму аутентифікації.

В якості об`єктно-орієнтованої мови програмування може бути обрані такі як, Delphi, C++, С#, Visual Basic . Вибір інших мов недоцільний.

Системні програмні засоби, що використовуються програмою, повинні бути представлені ліцензійною версією операційної системи Windows ХР або вище, MS Office, Firebird 2.5 або вище.

У процесі роботи програми вхідний інформацією для програми повинні бути: файли баз даних, маніпуляції мишею, а також інформація, що вводиться користувачем на клавіатурі ЕОМ, згідно режимам, що визначаються вихідний екранної інформацією.

Так як інформаційна система буде багатокористувацькою, то в ній будуть використовуватися розмежування доступу до системи, а вхід до системи буде відбуватися за паролем.

### 2.6.2 Структура мережної системи (топологічна і логічна)

В основі проектування АС лежить клієнт-серверна архітектура.

Клієнт-сервер являє собою обчислювальну або мережеву архітектуру, в якій завдання або мережева навантаження розподілені між постачальниками послуг, які називаються серверами, і замовниками послуг, званих клієнтами. Нерідко клієнти і сервери взаємодіють через комп'ютерну мережу і можуть бути як різними фізичними пристроями, так і програмним забезпеченням.

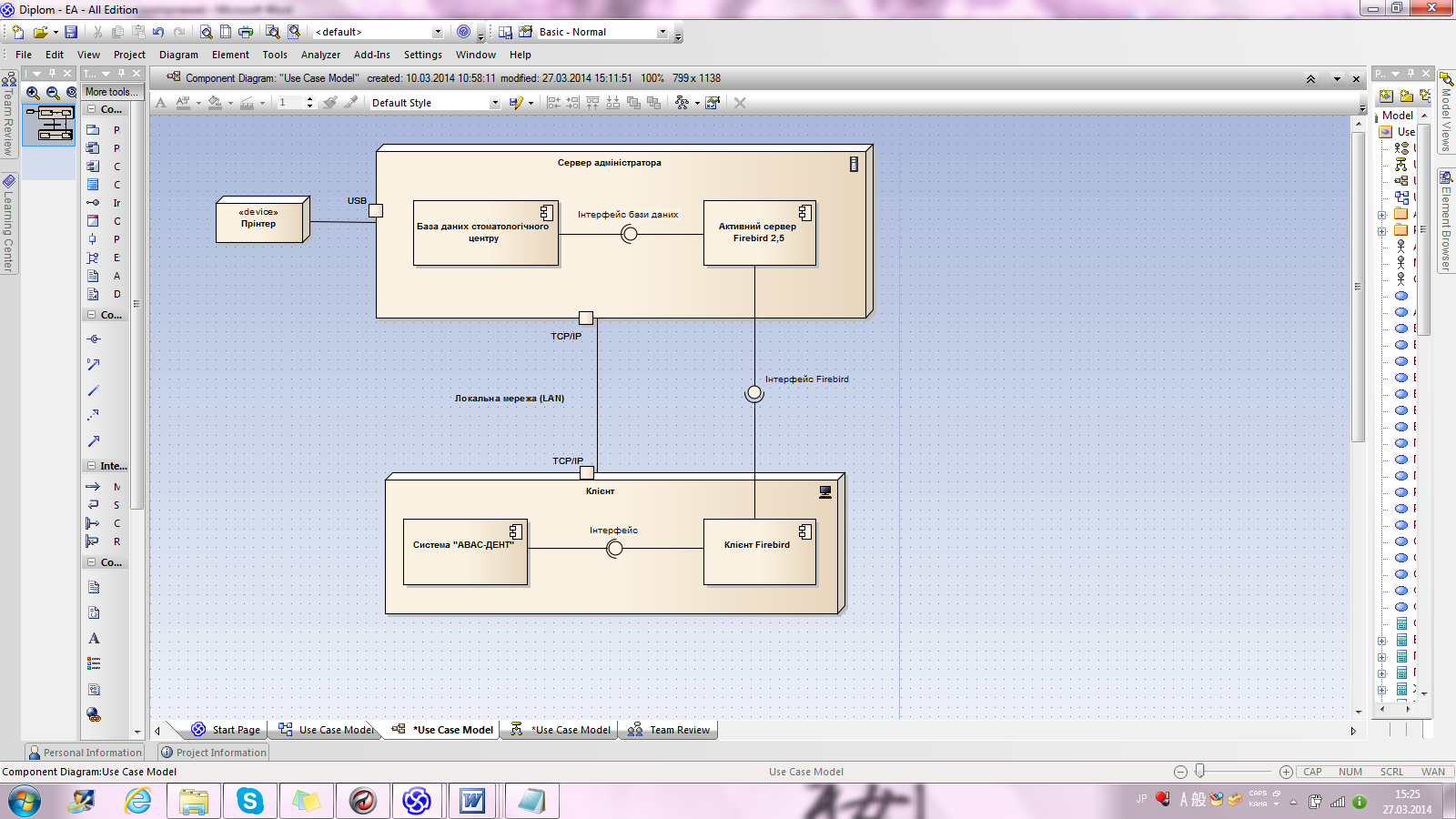


Рисунок 2.6.2 - Структура мережної системи (топологічна і логічна)

### 2.6.3 Структура комплексу засобів автоматизації

Комплекс засобів автоматизації (КСА), виконує ряд функцій по обробці інформації. Система включає в себе часткову автоматизацію обліку роботи стоматолога та медсестрі.

Таку, як, розробка системи, яка допоможе адміністратору стоматологічного центру в обробці, пошуку, зберіганні інформації про пацієнтів стоматологічного центру, розкладі прийомів і роботи лікарів. Допомога фахівцю - стоматологу у веденні медичної карти пацієнта. Допомога медсестрі у веденні обліку матеріалів та інструментарію, а також інформації про лікування хвороби пацієнта. Підвищення ефективності роботи всіх підрозділів стоматологічного центру та ведення обліку в єдиній інформаційній системі. АІС реалізовано на одному ПК

### 2.6.4 Специфікація обладнання

Вимоги до сервера: процесор Intel Core, мінімальний обсяг оперативної пам'яті 2 Гб, мінімальний обсяг дискового простору 60 ГБ, операційна система Windows XP/Vista/7. Вимоги до клієнтської машині: процесор Intel Core, мінімальний обсяг оперативної пам'яті 2 ГБ, мінімальний обсяг дискового простору 60 ГБ, операційна система Windows XP/Vista/7. Та периферійний пристрій – принтер.

# Висновок

Використання ЕОМ у різних сферах діяльності, багато в чому полегшує працю людини. XX століття принесло в життя людей багато досягнень і відкриттів. І майже все це відбулося завдяки обчислювальній техніці. Від заповнення сотень нікому не потрібних бланків, ми поступово переходимо до без паперової інформаційної технології управління. Система може працювати досить стабільно, була правильно обрана СУБД, що гарантує функціональність і безпеку даних.

Була розроблена автоматизована інформаційна система обліку роботи стоматологічного центру «АВАС-ДЕНТ». Вона допоможе адміністратору центру в обробці, пошуку, зберіганні інформації про пацієнтів центру, розкладі прийомів і роботи лікарів. Допоможе фахівцю - стоматологу у веденні медичної карти пацієнта, а так само медсестрі у веденні обліку матеріалів, інструментарію, та введення інформації про лікування хвороб пацієнта. Програма має наочний вид, легка у використанні, дані після закінчення роботи користувача оновлюються. Перевагою такої програми є і те, що вся інформація зберігається в комп'ютері.

Була детально розглянута діяльність стоматологічного центру, наведена інформація про підприємство, описані підрозділи та користувачі. Проведений аналіз існуючого рівня автоматизації та детально описані бізнес-процеси з наведеними таблицями дають змогу для подальшого модулювання проекту.

При розробці проекту були наведені основні методи та засоби структурного системного аналізу та проектування. В даному розділі були виконанні такі моделі, як: контекстна діаграма за методологію IDEF0, в якій були розглянуті основні бізнес – процеси стоматологічного центру як-є і як- буде, та декомпозиції для більш детального поглиблення і подальшого модулювання.

Функціональна модель системи за методологією IDEF3 розглядає бізнес процеси в іншій методології для всебічного огляду та подальшого модулювання проекту. Моделі потоків даних та діаграми дерева вузлів, FEO-діаграми показують структуру бізнес процесів в майбутньому проекті .

Інформаційна модель системи за методологією IDEF1X розглядає сутності та взаємозв’язки між ними, з яких складається база даних для проекту.

Поведінкова модель системи за методологією Flow Chart, BPMN, EPC деталізують бізнес процеси системи стоматологічного центру для автоматизованої системи.

АІС надзвичайно проста в експлуатації, не вимагає спеціальної кваліфікації персоналу, і розрахована на постійне використання персоналом стоматологічного центру.

Користувачами інформаційної системи є стоматолог, адміністратор, медсестра стоматологічного центру.

Розроблено зручний і функціональний інтерфейс, використані різні довідкові матеріали, а так само деякі нестандартні компоненти, програма наповнена різноманітними функціями, вона може реально використовуватися, підвищувати ефективність управління і знизити збитки фірми. Крім того, проект може бути розширений і на підставі його можливе створення потужної АС для масового використання.

При написанні дипломного проекту враховувалися концептуальні вимоги до системи. Була розроблена автоматизована інформаційна система обліку роботи стоматологічного центру «АВАС-ДЕНТ». Вона допоможе адміністратору центру в обробці, пошуку, зберіганні інформації про пацієнтів центру, розкладі прийомів і роботи лікарів. Допоможе фахівцю - стоматологу у веденні медичної карти пацієнта, а так само медсестрі у веденні обліку матеріалів, інструментарію, та введення інформації про лікування хвороб пацієнта. Програма має наочний вид, легка у використанні, дані після закінчення роботи користувача оновлюються. Перевагою такої програми є і те, що вся інформація зберігається в комп'ютері.

При впровадженні нової системи, необхідно враховувати не тільки її гнучкість і легкість в експлуатації, але і вартісні дані.

Отже, підводячи підсумок дипломного проекту можна сміливо сказати, що поставлені цілі були виконані.

# Перелік використаних джерел

1. Борри Х. Firebird: керівництво розроблювача баз даннях. - СПб.: БХВ- Петербург, 2007 р. - 1104 с.
2. Буч Г., Рамбо Дж., Джекобсон А. Мова UML. Керівництво користувача-М.: ДМК, 2000. - 432 с.
3. Грабер М. SQL. - М,: Лорі, 2003. - 672 с.
4. Джордан Д. Обробка об'єктних баз даних в C + +. Програмування з використанням стандарту ODMG. - М,: Вільямс, 2001 р.
5. Липа В.В. Програмна інженерія. Методологічні основи. - М.: Видавництво «Теис», 2006. - 316 с.
6. Структура и правила оформлення та Стандарту підприємства (Загальні вимоги до Виконання та оформлення текстових документів) Маріупольського механіко-металургійного коледжу ДВНЗ «ПДТУ», 2011р.
7. Шилдт, Герберт. С++ Базовый курс, 3-е издание. : Пер. с англ. – Алексей.
8. Грабер М. SQL. - М,: Лорі, 2003. - 672 с.
9. Липа В.В. Програмна інженерія. Методологічні основи. - М.: Видавництво «Теис», 2006. - 316 с.
10. Структура и правила оформлення та Стандарту підприємства (Загальні вимоги до Виконання та оформлення текстових документів) Маріупольського механіко-металургійного коледжу ДВНЗ «ПДТУ», 2011р.
11. ДСТУ 3008-95. Документація. Звіти у сфері науки і техніки. Структура і правила оформлення. – К.: Держстандарт України, 1995. – 38с.
12. INTEGRATION DEFINITION FOR FUNCTION MODELING (IDEF0) . Draft Federal Information Processing Standards Publication 183 ,1993 December 21
13. INTEGRATION DEFINITION FOR INFORMATION MODELING (IDEF1X), Draft Federal Information Processing Standards Publication 184 1993 December 21.
14. Давид Марка, Клемент МакГоуэн, Методология структурного анализа и проектирования. Пер. с англ. М.:1993, 240 с. , ISBN 5-7395-0007-9
15. Дружинин В.В., Конторов Д.С. Системотехника. – М.: Радио и связь. 1985, - 200 с.

# Додаток А DDL-скрипт

CREATE TABLE AVT (

UK\_AVT INTEGER NOT NULL,

SP VARCHAR(20),

PAR VARCHAR(20),

UR INTEGER);

ALTER TABLE AVT ADD PRIMARY KEY (UK\_AVT) USING INDEX RDB$PRIMARY15;

SET TERM ^ ;

CREATE TRIGGER BI\_AVT\_UK\_AVT FOR AVT

ACTIVE BEFORE INSERT

POSITION 0

AS

BEGIN

IF (NEW.UK\_AVT IS NULL) THEN

NEW.UK\_AVT = GEN\_ID(AVT\_UK\_AVT\_GEN, 1);

END^

SET TERM ; ^

CREATE TABLE GR\_PR\_P (

UK\_GR\_PR\_P INTEGER NOT NULL,

UK\_MK INTEGER,

UK\_PERS INTEGER,

DATA\_P DATE,

VREM TIME,

DIAGN VARCHAR(100),

PRIG VARCHAR(100),

JALOB VARCHAR(100),

LECH VARCHAR(400),

REZ VARCHAR(55));

ALTER TABLE GR\_PR\_P ADD PRIMARY KEY (UK\_GR\_PR\_P) USING INDEX RDB$PRIMARY2;

ALTER TABLE GR\_PR\_P ADD CONSTRAINT FK\_GR\_PR\_P FOREIGN KEY (UK\_PERS) REFERENCES PERSONAL(UK\_PERS);

ALTER TABLE GR\_PR\_P ADD CONSTRAINT FK\_GR\_PR\_P1 FOREIGN KEY (UK\_MK) REFERENCES MK(UK\_MK);

SET TERM ^ ;

CREATE TRIGGER BI\_GR\_PR\_P\_UK\_GR\_PR\_P FOR GR\_PR\_P

ACTIVE BEFORE INSERT

POSITION 0

AS

BEGIN

IF (NEW.UK\_GR\_PR\_P IS NULL) THEN

NEW.UK\_GR\_PR\_P = GEN\_ID(GR\_PR\_P\_UK\_GR\_PR\_P\_GEN, 1);

END^

SET TERM ; ^

CREATE TABLE GR\_R\_LL (

UK\_GR\_R\_L INTEGER NOT NULL,

UK\_PERS INTEGER,

DNI\_PR VARCHAR(25),

VR\_NACH TIME,

VR\_OKOC TIME);

ALTER TABLE GR\_R\_LL ADD PRIMARY KEY (UK\_GR\_R\_L) USING INDEX RDB$PRIMARY8;

ALTER TABLE GR\_R\_LL ADD CONSTRAINT FK\_GR\_R\_LL FOREIGN KEY (UK\_PERS) REFERENCES PERSONAL(UK\_PERS);

SET TERM ^ ;

CREATE TRIGGER BI\_GR\_R\_LL\_UK\_GR\_R\_L FOR GR\_R\_LL

ACTIVE BEFORE INSERT

POSITION 0

AS

BEGIN

IF (NEW.UK\_GR\_R\_L IS NULL) THEN

NEW.UK\_GR\_R\_L = GEN\_ID(GR\_R\_LL\_UK\_GR\_R\_L\_GEN, 1);

END^

SET TERM ; ^

CREATE TABLE IS\_MATER (

UK\_IS\_MAT INTEGER NOT NULL,

UK\_GR\_PR\_P INTEGER,

UK\_MAT INTEGER,

KOL\_VO INTEGER);

ALTER TABLE IS\_MATER ADD PRIMARY KEY (UK\_IS\_MAT) USING INDEX RDB$PRIMARY6;

ALTER TABLE IS\_MATER ADD CONSTRAINT FK\_IS\_MATER FOREIGN KEY (UK\_MAT) REFERENCES MATER(UK\_MMM);

ALTER TABLE IS\_MATER ADD CONSTRAINT FK\_IS\_MATER1 FOREIGN KEY (UK\_GR\_PR\_P) REFERENCES GR\_PR\_P(UK\_GR\_PR\_P);

SET TERM ^ ;

CREATE TRIGGER BI\_IS\_MATER\_UK\_IS\_MAT FOR IS\_MATER

ACTIVE BEFORE INSERT

POSITION 0

AS

BEGIN

IF (NEW.UK\_IS\_MAT IS NULL) THEN

NEW.UK\_IS\_MAT = GEN\_ID(IS\_MATER\_UK\_IS\_MAT\_GEN, 1);

END^

SET TERM ; ^

CREATE TABLE IS\_POSL (

UK\_IS\_POSL INTEGER NOT NULL,

UK\_GR\_PR\_P INTEGER,

UK\_PRE INTEGER);

ALTER TABLE IS\_POSL ADD PRIMARY KEY (UK\_IS\_POSL) USING INDEX RDB$PRIMARY13;

ALTER TABLE IS\_POSL ADD CONSTRAINT FK\_IS\_POSL FOREIGN KEY (UK\_GR\_PR\_P) REFERENCES GR\_PR\_P(UK\_GR\_PR\_P);

ALTER TABLE IS\_POSL ADD CONSTRAINT FK\_IS\_POSL1 FOREIGN KEY (UK\_PRE) REFERENCES PRESC(UK\_PRESC);

SET TERM ^ ;

CREATE TRIGGER BI\_IS\_POSL\_UK\_IS\_POSL FOR IS\_POSL

ACTIVE BEFORE INSERT

POSITION 0

AS

BEGIN

IF (NEW.UK\_IS\_POSL IS NULL) THEN

NEW.UK\_IS\_POSL = GEN\_ID(IS\_POSL\_UK\_IS\_POSL\_GEN, 1);

END^

SET TERM ; ^

CREATE TABLE MATER (

UK\_MMM INTEGER NOT NULL,

ART INTEGER,

NAZVA VARCHAR(60),

TSINA FLOAT,

KOL INTEGER);

ALTER TABLE MATER ADD PRIMARY KEY (UK\_MMM) USING INDEX RDB$PRIMARY7;

SET TERM ^ ;

CREATE TRIGGER BI\_MATER\_UK\_MMM FOR MATER

ACTIVE BEFORE INSERT

POSITION 0

AS

BEGIN

IF (NEW.UK\_MMM IS NULL) THEN

NEW.UK\_MMM = GEN\_ID(MATER\_UK\_MMM\_GEN, 1);

END^

SET TERM ; ^

CREATE TABLE MK (

UK\_MK INTEGER NOT NULL,

PIB VARCHAR(25),

DATA\_R DATE,

OP\_Z\_P VARCHAR(100),

P\_S\_Z VARCHAR(100),

TEL VARCHAR(25),

ADRES VARCHAR(55),

ST VARCHAR(1),

N\_CH BLOB,

VV BLOB);

ALTER TABLE MK ADD PRIMARY KEY (UK\_MK) USING INDEX RDB$PRIMARY4;

SET TERM ^ ;

CREATE TRIGGER BI\_MK\_UK\_MK FOR MK

ACTIVE BEFORE INSERT

POSITION 0

AS

BEGIN

IF (NEW.UK\_MK IS NULL) THEN

NEW.UK\_MK = GEN\_ID(MK\_UK\_MK\_GEN, 1);

END^

SET TERM ; ^

CREATE TABLE PERSONAL (

UK\_PERS INTEGER NOT NULL,

PIB\_PERS VARCHAR(55),

SPES VARCHAR(55),

AD VARCHAR(55),

TELEFON INTEGER);

ALTER TABLE PERSONAL ADD PRIMARY KEY (UK\_PERS) USING INDEX RDB$PRIMARY5;

SET TERM ^ ;

CREATE TRIGGER BI\_PERSONAL\_UK\_PERS FOR PERSONAL

ACTIVE BEFORE INSERT

POSITION 0

AS

BEGIN

IF (NEW.UK\_PERS IS NULL) THEN

NEW.UK\_PERS = GEN\_ID(PERSONAL\_UK\_PERS\_GEN, 1);

END^

SET TERM ; ^

CREATE TABLE POST\_MAT (

UK\_POST\_MAT INTEGER NOT NULL,

UK\_MMM INTEGER,

KOL\_VO INTEGER,

DATA DATE,

KIOL\_POST INTEGER,

DATA\_POS DATE,

PRICH VARCHAR(55));

ALTER TABLE POST\_MAT ADD PRIMARY KEY (UK\_POST\_MAT) USING INDEX RDB$PRIMARY10;

ALTER TABLE POST\_MAT ADD CONSTRAINT FK\_POST\_MAT FOREIGN KEY (UK\_MMM) REFERENCES MATER(UK\_MMM);

SET TERM ^ ;

CREATE TRIGGER BI\_POST\_MAT\_UK\_POST\_MAT FOR POST\_MAT

ACTIVE BEFORE INSERT

POSITION 0

AS

BEGIN

IF (NEW.UK\_POST\_MAT IS NULL) THEN

NEW.UK\_POST\_MAT = GEN\_ID(POST\_MAT\_UK\_POST\_MAT\_GEN, 1);

END^

SET TERM ; ^

CREATE TABLE PRESC (

UK\_PRESC INTEGER NOT NULL,

NAZ VARCHAR(25),

TS FLOAT);

ALTER TABLE PRESC ADD PRIMARY KEY (UK\_PRESC) USING INDEX RDB$PRIMARY9;

SET TERM ^ ;

CREATE TRIGGER BI\_PRESC\_UK\_PRESC FOR PRESC

ACTIVE BEFORE INSERT

POSITION 0

AS

BEGIN

IF (NEW.UK\_PRESC IS NULL) THEN

NEW.UK\_PRESC = GEN\_ID(PRESC\_UK\_PRESC\_GEN, 1);

END^

SET TERM ; ^

CREATE TABLE ZUB\_KARTA (

UK\_Z INTEGER NOT NULL,

UK\_MK INTEGER,

SOST VARCHAR(100),

NOM\_Z VARCHAR(10));

ALTER TABLE ZUB\_KARTA ADD PRIMARY KEY (UK\_Z) USING INDEX RDB$PRIMARY12;

ALTER TABLE ZUB\_KARTA ADD CONSTRAINT FK\_ZUB\_KARTA FOREIGN KEY (UK\_MK) REFERENCES MK(UK\_MK);

SET TERM ^ ;

CREATE TRIGGER BI\_ZUB\_KARTA\_UK\_Z FOR ZUB\_KARTA

ACTIVE BEFORE INSERT

POSITION 0

AS

BEGIN

IF (NEW.UK\_Z IS NULL) THEN

NEW.UK\_Z = GEN\_ID(ZUB\_KARTA\_UK\_Z\_GEN, 1);

END^

SET TERM ; ^

Додаток А Лістинг програми

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

#include "Unit2.h"

#include "Project1PCH1.h"

#include "Unit3.h"

#include "Unit4.h"

#include "Unit5.h"

#include "Unit7.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm1 \*Form1;

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Button1Click(TObject \*Sender)

{

TLocateOptions SearchOptions;

dm->AVT->Locate("SP",Edit1->Text,SearchOptions<<loPartialKey<<loCaseInsensitive);

if(dm->AVTPAR->Value==Edit2->Text) {

if (dm->AVTUR->Value==4)

{

Form2->PageControl1->Pages[9]->TabVisible=false;

Form2->PageControl1->Pages[2]->TabVisible=false;

Form2->PageControl1->Pages[0]->TabVisible=false;

Form2->Show();}

if (dm->AVTUR->Value==20)

{

Form2->PageControl1->Pages[0]->TabVisible=false;

Form2->PageControl1->Pages[7]->TabVisible=false;

Form2->PageControl1->Pages[2]->TabVisible=false;

Form2->PageControl1->Pages[5]->TabVisible=false;

Form2->Show();}

if (dm->AVTUR->Value==14)

{

Form2->Show();}

if(dm->AVTUR->Value==1)

{

Form2->PageControl1->Pages[5]->TabVisible=false;

Form2->PageControl1->Pages[7]->TabVisible=false;

Form2->Show(); }

}

else ShowMessage("Неверно введена специальность или пароль!");

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "utilcls.h"

#include "sysvari.h"

#include "ComObj.hpp"

#include "Unit1.h"

#include "Unit2.h"

#include "Unit3.h"

#include "Unit4.h"

#include "Unit5.h"

#include "Project1PCH1.h"

#include "Unit7.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm2 \*Form2;

AnsiString sql;

//---------------------------------------------------------------------------

\_\_fastcall TForm2::TForm2(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::N3Click(TObject \*Sender)

{ Form2->Close();

Form1->Show();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button3Click(TObject \*Sender)

{ if(dm->AVTUR->Value==4) //med

{Form3->Edit1->Visible=false;

Form3->Edit7->Visible=false;

Form3->Label5->Visible=false;

Form3->DateTimePicker1->Visible=false;

Form3->Edit3->Visible=false;

Form3->Edit4->Visible=false;

Form3->Edit2->Visible=false;

Form3->Edit6->Visible=true;

Form3->Edit8->Visible=true;

Form3->Label3->Visible=true;

Form3->Label4->Visible=true;

Form3->DateTimePicker4->Visible=true;

Form3->DateTimePicker3->Visible=true;

Form3->Label2->Visible=false;

Form3->DateTimePicker2->Visible=false;

Form3->Show();}

if(dm->AVTUR->Value==1) //admin

{Form3->Edit1->Visible=true;

Form3->Edit7->Visible=true;

Form3->Edit2->Visible=true;

Form3->Edit2->Visible=true;

Form3->Label5->Visible=true;

Form3->DateTimePicker1->Visible=true;

Form3->Edit3->Visible=false;

Form3->Edit4->Visible=false;

Form3->Label2->Visible=false;

Form3->DateTimePicker2->Visible=false;

Form3->Edit6->Visible=false;

Form3->Edit8->Visible=false;

Form3->Label3->Visible=false;

Form3->Label4->Visible=false;

Form3->DateTimePicker4->Visible=false;

Form3->DateTimePicker3->Visible=false;

Form3->Show(); }

if(dm->AVTUR->Value==20) //st

{Form3->Edit1->Visible=false;

Form3->Edit7->Visible=false;

Form3->Label5->Visible=false;

Form3->DateTimePicker1->Visible=false;

Form3->Edit2->Visible=false;

Form3->Edit3->Visible=true;

Form3->Edit4->Visible=true;

Form3->Label2->Visible=true;

Form3->DateTimePicker2->Visible=true;

Form3->Edit6->Visible=false;

Form3->Edit8->Visible=false;

Form3->Label3->Visible=false;

Form3->Label4->Visible=false;

Form3->DateTimePicker4->Visible=false;

Form3->DateTimePicker3->Visible=false;

Form3->Show(); }

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button5Click(TObject \*Sender)

{

Form4->Show();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::GroupBox2MouseUp(TObject \*Sender, TMouseButton Button, TShiftState Shift,

int X, int Y)

{

;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::RadioButton1Click(TObject \*Sender)

{

if(RadioButton1->Checked)

DateTimePicker1->Visible=true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::RadioButton2Click(TObject \*Sender)

{

if(RadioButton2->Checked)

Edit6->Visible=false;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::FormMouseDown(TObject \*Sender, TMouseButton Button, TShiftState Shift,

int X, int Y)

{

long SC\_DRAGMOVE = 0xF012;

if(Button == mbLeft)

{

ReleaseCapture();

SendMessage(Handle, WM\_SYSCOMMAND, SC\_DRAGMOVE, 0);

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::RadioButton3Click(TObject \*Sender)

{

if(RadioButton3->Checked)

DateTimePicker5->Visible=false;

DateTimePicker6->Visible=false;

Edit1->Visible=true;

Edit8->Visible=true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Edit1Change(TObject \*Sender)

{

if(RadioButton3->Checked)

//DateTimePicker3->Visible=true;

//DateTimePicker4->Visible=true;

Edit1->Visible=false;

Edit8->Visible=false;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::RadioButton9Click(TObject \*Sender)

{

Edit13->Visible=false;

Edit14->Visible=false;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::FormClose(TObject \*Sender, TCloseAction &Action)

{

Form1->Close();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::RadioButton11Click(TObject \*Sender)

{

Edit13->Visible=true;

Edit14->Visible=true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::RadioButton4Click(TObject \*Sender)

{ DateTimePicker5->Visible=true;

DateTimePicker6->Visible=true;

Edit1->Visible=false;

Edit8->Visible=false;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::RadioButton17Click(TObject \*Sender)

{

DateTimePicker7->Visible=true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::RadioButton18Click(TObject \*Sender)

{

Edit6->Visible=false;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::RadioButton19Click(TObject \*Sender)

{

DateTimePicker9->Visible=true;

DateTimePicker10->Visible=true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::RadioButton14Click(TObject \*Sender)

{

ComboBox3->Visible=true;

ComboBox4->Visible=false;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::RadioButton15Click(TObject \*Sender)

{

ComboBox3->Visible=false;

ComboBox4->Visible=true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::TabSheet12ContextPopup(TObject \*Sender, TPoint &MousePos,

bool &Handled)

{ TabSheet12->TabVisible = false;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button7Click(TObject \*Sender)

{

dm->GR\_PP->Filtered=true;

dm->PERS->Filtered=true;

dm->POST\_MAT->Filtered=true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button6Click(TObject \*Sender)

{

Form5->Show();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::DBNavigator2Click(TObject \*Sender, TNavigateBtn Button)

{

dm->GR\_PR\_P->Active=false;

dm->GR\_PR\_P->ParamByName("kod")->Value=dm->MKUK\_MK->Value;

dm->GR\_PR\_P->Active=true;

dm->IS\_MAT->Active=false;

dm->IS\_MAT->ParamByName("kod")->Value=dm->GR\_PR\_PUK\_GR\_PR\_P->Value;

dm->IS\_MAT->Active=true;

dm->IS\_POSL->Active=false;

dm->IS\_POSL->ParamByName("kod")->Value=dm->GR\_PR\_PUK\_GR\_PR\_P->Value;

dm->IS\_POSL->Active=true;

dm->ZUB->Active=false;

dm->ZUB->ParamByName("kod")->Value=dm->MKUK\_MK->Value;

dm->ZUB->Active=true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::FormCreate(TObject \*Sender)

{

dm->GR\_PR\_P->Active=false;

dm->GR\_PR\_P->ParamByName("kod")->Value=dm->MKUK\_MK->Value;

dm->GR\_PR\_P->Active=true;

dm->IS\_MAT->Active=false;

dm->IS\_MAT->ParamByName("kod")->Value=dm->GR\_PR\_PUK\_GR\_PR\_P->Value;

dm->IS\_MAT->Active=true;

dm->IS\_POSL->Active=false;

dm->IS\_POSL->ParamByName("kod")->Value=dm->GR\_PR\_PUK\_GR\_PR\_P->Value;

dm->IS\_POSL->Active=true;

dm->GR\_R\_L->Active=false;

dm->GR\_R\_L->ParamByName("kod")->Value=dm->PERSUK\_PERS->Value;

dm->GR\_R\_L->Active=true;

dm->ZUB->Active=false;

dm->ZUB->ParamByName("kod")->Value=dm->MKUK\_MK->Value;

dm->ZUB->Active=true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::DBNavigator12Click(TObject \*Sender, TNavigateBtn Button)

{

dm->IS\_MAT->Active=false;

dm->IS\_MAT->ParamByName("kod")->Value=dm->GR\_PR\_PUK\_GR\_PR\_P->Value;

dm->IS\_MAT->Active=true;

dm->IS\_POSL->Active=false;

dm->IS\_POSL->ParamByName("kod")->Value=dm->GR\_PR\_PUK\_GR\_PR\_P->Value;

dm->IS\_POSL->Active=true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::DBGrid12CellClick(TColumn \*Column)

{

dm->IS\_MAT->Active=false;

dm->IS\_MAT->ParamByName("kod")->Value=dm->GR\_PR\_PUK\_GR\_PR\_P->Value;

dm->IS\_MAT->Active=true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::DBNavigator10Click(TObject \*Sender, TNavigateBtn Button)

{

dm->GR\_R\_L->Active=false;

dm->GR\_R\_L->ParamByName("kod")->Value=dm->PERSUK\_PERS->Value;

dm->GR\_R\_L->Active=true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::DBGrid14CellClick(TColumn \*Column)

{

dm->GR\_R\_L->Active=false;

dm->GR\_R\_L->ParamByName("kod")->Value=dm->PERSUK\_PERS->Value;

dm->GR\_R\_L->Active=true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::DBGrid15CellClick(TColumn \*Column)

{

dm->IS\_POSL->Active=false;

dm->IS\_POSL->ParamByName("kod")->Value=dm->GR\_PR\_PUK\_GR\_PR\_P->Value;

dm->IS\_POSL->Active=true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::DBGrid1CellClick(TColumn \*Column)

{

dm->ZUB->Active=false;

dm->ZUB->ParamByName("kod")->Value=dm->MKUK\_MK->Value;

dm->ZUB->Active=true;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button8Click(TObject \*Sender)

{

dm->GR\_PP->Filtered=false;

dm->PERS->Filtered=false;

dm->POST\_MAT->Filtered=false;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button4Click(TObject \*Sender)

{

Variant mExcel;

AnsiString sFile = GetCurrentDir()+"\\Чек про оплату.xls";

AnsiString s1;

int i=1;

mExcel=CreateOleObject("Excel.Application"); //Запускаем экземпляр Excel

mExcel.OlePropertySet("Visible", True); //Делаем экземпляр видимым

mExcel.OlePropertyGet("WorkBooks").OleProcedure("Open",sFile.c\_str(),0,false,1);

s1="№ чеку"+IntToStr(dm->MKUK\_MK->Value);

mExcel.OlePropertyGet("ActiveSheet").OlePropertyGet("Cells", i, "A").OlePropertySet("Value",s1.c\_str());

}

s1="ПІБ пацієнта"+(dm->MKPIB->Value);

mExcel.OlePropertyGet("ActiveSheet").OlePropertyGet("Cells", i+1, "A").OlePropertySet("Value",s1.c\_str());

i=i+2;

s1="ПІБ лікаря"; //+(dm->PERSPIB\_PERS->Value);

mExcel.OlePropertyGet("ActiveSheet").OlePropertyGet("Cells", i, "A").OlePropertySet("Value",s1.c\_str());

s1="Дата прийому";//+DateToStr( dm->GR\_PR\_PDATA\_P->Value);

mExcel.OlePropertyGet("ActiveSheet").OlePropertyGet("Cells", i, "A").OlePropertySet("Value",s1.c\_str());

s1="Діагноз";//+(dm->GR\_PR\_PDIAGN->Value;);

mExcel.OlePropertyGet("ActiveSheet").OlePropertyGet("Cells", i, "A").OlePropertySet("Value",s1.c\_str());

s1="Лікування";//+(dm->GR\_PR\_PLECH->Value;);

mExcel.OlePropertyGet("ActiveSheet").OlePropertyGet("Cells", i, "A").OlePropertySet("Value",s1.c\_str());

while(!dm->GR\_PR\_P->Eof)

{ i++;

mExcel.OlePropertyGet("ActiveSheet").OlePropertyGet("Cells", i, "A").OlePropertySet("Value",(dm->GR\_PR\_Pp->AsAnsiString).c\_str());

mExcel.OlePropertyGet("ActiveSheet").OlePropertyGet("Cells", i, "A").OlePropertySet("Value", (dm->GR\_PR\_PDATA\_P->AsAnsiString).c\_str());

mExcel.OlePropertyGet("ActiveSheet").OlePropertyGet("Cells", i, "A").OlePropertySet("Value", (dm->GR\_PR\_PDIAGN->AsAnsiString).c\_str());

mExcel.OlePropertyGet("ActiveSheet").OlePropertyGet("Cells", i, "A").OlePropertySet("Value", (dm->GR\_PR\_PLECH->AsAnsiString).c\_str());

());

dm->GR\_PR\_P->Next();

}

i++;

s1="Назва послуги";//+(dm->PRESCNAZ->Value;);

mExcel.OlePropertyGet("ActiveSheet").OlePropertyGet("Cells", i, "A").OlePropertySet("Value",s1.c\_str());

s1="Сума,грн";//+(dm->PRESCTS->Value;);

mExcel.OlePropertyGet("ActiveSheet").OlePropertyGet("Cells", i, "A").OlePropertySet("Value",s1.c\_str());

while(!dm->PRESC->Eof)

{ i++;

mExcel.OlePropertyGet("ActiveSheet").OlePropertyGet("Cells", i, "A").OlePropertySet("Value",(dm->PRESCNAZ->AsAnsiString).c\_str());

mExcel.OlePropertyGet("ActiveSheet").OlePropertyGet("Cells", i, "A").OlePropertySet("Value", (dm->PRESCTS->AsAnsiString).c\_str());

dm->PRESC->Next();

}

i++;

s1="Cума до сплати,грн "+(this->Label26->Caption);

mExcel.OlePropertyGet("ActiveSheet").OlePropertyGet("Cells", i, "A").OlePropertySet("Value",s1.c\_str());

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit3.h"

#include "Project1PCH1.h"

#include "Unit1.h"

#include "Unit2.h"

#include "Unit4.h"

#include "Unit5.h"

#include "Unit7.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm3 \*Form3;

int i;

//---------------------------------------------------------------------------

\_\_fastcall TForm3::TForm3(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::Button1Click(TObject \*Sender)

{

TLocateOptions SearchOptions;

dm->MK->Locate("PIB",this->Edit1->Text,SearchOptions<<loPartialKey<<loCaseInsensitive); //admin

dm->PERS->Locate("PIB\_PERS",this->Edit7->Text,SearchOptions<<loPartialKey<<loCaseInsensitive);

dm->PRESC->Locate("NAZ",this->Edit2->Text,SearchOptions<<loPartialKey<<loCaseInsensitive);

dm->MK->Locate("PIB",this->Edit3->Text,SearchOptions<<loPartialKey<<loCaseInsensitive); //st

dm->PERS->Locate("PIB\_PERS",this->Edit4->Text,SearchOptions<<loPartialKey<<loCaseInsensitive);

dm->MK->Locate("PIB",this->Edit8->Text,SearchOptions<<loPartialKey<<loCaseInsensitive); //med

dm->REES\_MAT->Locate("NAZVA",this->Edit6->Text,SearchOptions<<loPartialKey<<loCaseInsensitive);

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit2.h"

#include "Unit4.h"

#include "Unit1.h"

#include "Unit3.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm4 \*Form4;

//---------------------------------------------------------------------------

\_\_fastcall TForm4::TForm4(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm4::Button1Click(TObject \*Sender)

{

Form2->Close();

Form3->Close();

Form1->Close();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm4::Button2Click(TObject \*Sender)

{

Form4->Close();

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit5.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm5 \*Form5;

//---------------------------------------------------------------------------

\_\_fastcall TForm5::TForm5(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

#pragma hdrstop

#include "Unit7.h"

#include "Project1PCH1.h"

#include "Unit1.h"

#include "Unit2.h"

#include "Unit3.h"

#include "Unit4.h"

#include "Unit5.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma classgroup "Vcl.Controls.TControl"

#pragma resource "\*.dfm"

Tdm \*dm;

//---------------------------------------------------------------------------

\_\_fastcall Tdm::Tdm(TComponent\* Owner)

: TDataModule(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall Tdm::GR\_PR\_PFilterRecord(TDataSet \*DataSet, bool &Accept)

{

unsigned short y,d,m;

this->DateTimePicker1->Date.DecodeDate(&y,&m,&d);

TLocateOptions SearchOptions;

dm->GR\_PP->Locate("DATA\_P",EncodeDate(y,m,d), SearchOptions«loPartialKey«loCaseInsensitive);

}

//---------------------------------------------------------------------------

void \_\_fastcall Tdm::POST\_MATFilterRecord(TDataSet \*DataSet, bool &Accept)

{

unsigned short o,p,u;

this->DateTimePicker1->Date.DecodeDate(&o,&p,&u);

TLocateOptions SearchOptions;

dm->GR\_PP->Locate("DATA\_POS",EncodeDate(o,p,u), SearchOptions«loPartialKey«loCaseInsensitive);

unsigned short l,h,f;

this->DateTimePicker1->Date.DecodeDate(&l,&h,&f);

TLocateOptions SearchOptions;

dm->GR\_PP->Locate("DATA\_POS",EncodeDate(l,h,f), SearchOptions«loPartialKey«loCaseInsensitive);

}

//---------------------------------------------------------------------------

void \_\_fastcall Tdm::PERSFilterRecord(TDataSet \*DataSet, bool &Accept)

{

if (Form2->RadioButton26->Checked) {

Accept=dm->PERSPIB\_PERS->Value==Form2->Edit11->Text; }

}

//---------------------------------------------------------------------------

void \_\_fastcall Tdm::GR\_R\_LBeforePost(TDataSet \*DataSet)

{

dm->GR\_R\_LUK\_PERS->Value=dm->PERSUK\_PERS->Value;

}

//---------------------------------------------------------------------------

void \_\_fastcall Tdm::ZUBBeforePost(TDataSet \*DataSet)

{ dm->ZUBUK\_MK->Value=dm->MKUK\_MK->Value;

}

//---------------------------------------------------------------------------

void \_\_fastcall Tdm::GR\_PR\_PBeforePost(TDataSet \*DataSet)

{

dm->GR\_PR\_PUK\_MK->Value=dm->MKUK\_MK->Value;

}

//---------------------------------------------------------------------------

void \_\_fastcall Tdm::IS\_MATBeforePost(TDataSet \*DataSet)

{

dm->IS\_POSLUK\_GR\_PR\_P->Value=dm->GR\_PR\_PUK\_GR\_PR\_P->Value;

}

//---------------------------------------------------------------------------

void \_\_fastcall Tdm::IS\_POSLBeforePost(TDataSet \*DataSet)

{

dm->IS\_MATUK\_GR\_PR\_P->Value=dm->GR\_PR\_PUK\_GR\_PR\_P->Value;

}

//---------------------------------------------------------------------------

void \_\_fastcall Tdm::REES\_MATFilterRecord(TDataSet \*DataSet, bool &Accept)

{

unsigned short x,b,c;

this->DateTimePicker2->Date.DecodeDate(&x,&b,&c);

TLocateOptions SearchOptions;

dm->REES\_MAT->Locate("KOL",EncodeDate(x,b,c), SearchOptions«loPartialKey«loCaseInsensitive)

}

//---------------------------------------------------------------------------