МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

ВИПУСКНА РОБОТА

на тему:

«Інформаційна система моніторингу співробітників підприємства ЕнергоАтом»

Завідувач

випускаючої кафедри **Довбиш А.С.**

Керівник роботи **Берест О.Б.**

Студента групи ІНдн-72с **Дериземля М.В.**

СУМИ 2021

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

Центр заочної, дистанційної і вечірньої форм навчання

Кафедра комп'ютерних наук

Затверджую_____

Зав. кафедрою Довбиш А.С.

"_____" _____2021 р.

ЗАВДАННЯ

до випускної роботи

Студента четвертого курсу, групи ІНдн-72с спеціальності "Комп'ютерні науки" дистанційної форми навчання Дериземлі Максима Віталійовича.

Тема: " Інформаційна система моніторингу співробітників підприємства ЕнергоАтом "

Затверджена наказом по СумДУ

№_____ від _____2021р.

Зміст пояснювальної записки: 1) коротка характеристика підприємства та середовища розробки; 2) технічне завдання; 3) аналіз моделі та предметної області; 4) структура та алгоритмізація бази даних;

Дата видачі завдання "_____"_____2021 р.

Керівник випускної роботи _____ Берест О.Б.

Завдання приняв до виконання _____ Дериземля М.В.

РЕФЕРАТ

Записка: 40 стор., 8 рис., 10 джерел.

Об’єкт дослідження — моніторинг пристроїв та співробітників підприємства.

Мета роботи — розробка програмного забезпечення (додатку) для співробітників підприємства, для моніторингу, додавання та сортування апаратного обладнання.

Методи дослідження — метод аналітично-прикладних випробувань.

Результати — Розглянуто корпоративний пакет програм, що містить інформацію про апаратне та програмне забезпечення. Розроблено підсистему обміну інформацією між двома клієнтськими додатками розподільної бази даних. Для підприємства створена зручна програма, за допомогою якої відділ інформаційних технологій може отримувати інформацію про доступне обладнанні і вимоги щодо його ремонту і технічного обслуговування.

МОНІТОРИНГ АПАРАТНОГО ОБЛАДНАННЯ, ІТ-ВІДДІЛ, ОПТИМІЗАЦІЯ РЕСУРСІВ, ЗБІР ДАНИХ, НАДІЙНІСТЬ ЗБЕРІГАННЯ ДАНИХ, АВТОМАТИЗОВАНА СИСТЕМА УПРАВЛІННЯ ІТ-ІНФРАСТРУКТУРОЮ, ОБ’ЄКТНА МОДЕЛЬ.

ЗМІСТ

ВСТУП…… 5
1 КОРОТКА ХАРАКТЕРИСТИКА ПІДПРИЄМСТВА ТА СЕРЕДОВИЩА РОЗРОБКИ…… 6
1.1 Коротка характеристика підприємства…….. 6
1.2 Аналіз об'єкта автоматизації. Структура ІТ-відділу……. 9
2 ТЕХНІЧНЕ ЗАВДАННЯ…….. 12
2.1 Найменування програми…….. 12
2.2 Призначення і область застосування……... 12
2.3 Вимоги до програми……. 12
2.4 Умови експлуатації……... 13
3 АНАЛІЗ МОДЕЛІ ТА ПРЕДМЕТНОЇ ОБЛАСТІ…….. 15
3.1 Опис предметної області…….. 15
3.2 Критичний опис існуючого програмного забезпечення…… 16
3.3 Об'єктна модель…… 17
3.4 Аналіз обраного середовища розробки та мови програмування…….. 18
4 СТРУКТУРА ТА АЛГОРИТМІЗАЦІЯ БАЗИ ДАНИХ…….. 21
4.1 Зразки бланків вихідних документів……... 21
4.2 Вибір інструментального середовища розробки…… 22
4.3 Постановка завдання…… 22
4.4 Загальні відомості про програмне забезпечення……... 23
4.5 Захист програмного продукту……. 34
4.6 Архітектура віддаленого опитування і занесення в БД……. 37
ВИСНОВКИ……... 39
СПИСОК ЛІТЕРАТУРИ…… 40

ВСТУП

Проаналізувавши наукові статті, можна зробити висновок, що інформаційні технології розвиваються дуже швидко. Ця швидкість справді вражає. Однак у комп’ютерному світі є одна з найважливіших сфер, і це те, що зміни відбуваються дуже повільно. Програмування, кодування, написання вихідного коду - ключові елементи створення будь-якої програми сьогодні такі ж, як і 40 років тому. Розробники використовують надзвичайно обмежений набір логічних конструкцій (умовні оператори та оператори присвоєння та циклу) та невелику кількість стандартних типів даних. Хоча було змінено більше покоління мов програмування, цей підхід не змінився зовсім. Наприклад, використовуючи C# замість C і Pascal, працівники різних комп'ютерних компаній все ще запускають компілятор командного рядка Java і налагоджують складні програми в консолі, переглядають роботу вручну та ігнорують швидкий візуальний розвиток. Зручно використовувати для налагодження, такі програми як JBuilder або NetBeans. Комп’ютерні видання, які стверджують, що мають професійні звання, часто пропонують подібні методи для створення програм. Створивши спеціальний образ, який може писати необхідні програми протягом декількох безсонних ночей, які вмістять 100 Кб пам'яті. Взагалі кажучи, програмування характеризується сильною консервативністю, оскільки в принципі ви можете створювати програми, які обмежені знаннями, що були набуті багато років тому. Однак у наш час програмування, безперечно, змінилося із мистецтва на ремесло. Звичайно, якщо ви не вивчите внутрішню структуру Windows або структуру компонентів VCL та принципи оптимізації програми, вам практично неможливо стати професійним розробником. Однак сьогодні цей вид знань поступово зникає.

Перед написанням дипломного проекту було проаналізовано масу інформації, включаючи аналіз структури IT-відділу, створення концепцій автоматизації та розробку сервісів для ведення баз даних.

1 КОРОТКА ХАРАКТЕРИСТИКА ПІДПРИЄМСТВА ТА СЕРЕДОВИЩА РОЗРОБКИ

1.1 Коротка характеристика підприємства.

ДП "ЕнергоАтом" було створено після реорганізації ВАТ "ЕнергоАтом" та зареєстровано як юридична особа 1 липня 2006 року. Реорганізація ВАТ "ЕнергоАтом" проводиться відповідно до основного напрямку національної політики реформування енергетики та проекту реформування компанії.

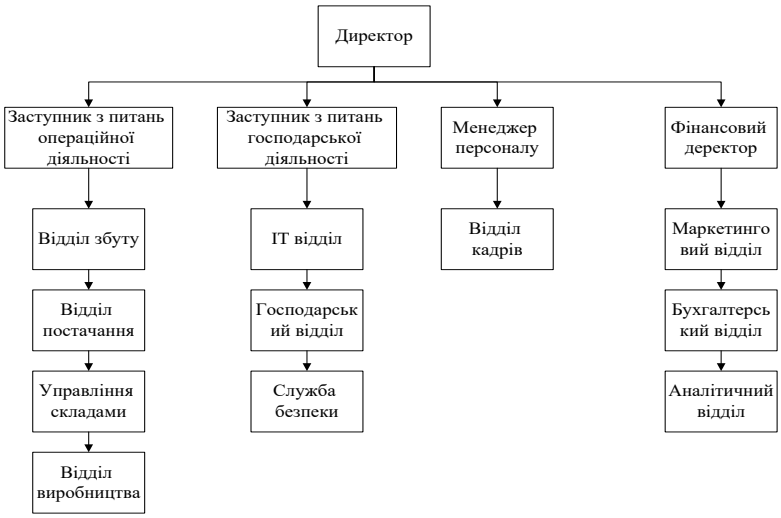

Рисунок 1 – Організаційна структура підприємства, яке було взято на аналіз

Державне підприємство "ЕнергоАтом" є найбільшим постачальником електроенергії для сільськогосподарських підприємств, промисловості, комерційних організацій та побутових споживачів по Україні[4].

Історія підприємства розпочалася в жовтні 1936 року, коли Народний комітет Радянського Союзу прийняв постанову про створення відділу збуту енергії в обласному енергетичному управлінні. У листопаді того ж року Регіональний департамент енергетики створив «Енергозбут енергоатом», до складу якого входять філії по всій Україні (охоплюють споживачів в енергетичних зонах країни). "Енергозбут" включає абонентські послуги, енергетичні інспекції, лабораторії та інші послуги. У 1946 році приймаючі відвідувачі запустили майстерню з ремонту лічильників. Федір Степанович Пономаренко стає першим директором енергозбутової компанії.

У 1980 р., Відповідно до вказівок Міністерства енергетики СРСР, компанія "Енергозбут" з питань продажу та контролю використання енергії була перейменована в "Енергонагляд", національну компанію з енергетичного нагляду та продажу енергії. У 1997 році філію акціонерного товариства "ЕнергоАтом" було перейменовано на "Енергозбут", а функція енергетичного нагляду передана державі. Відповідно до постанови уряду України "про реформування енергетики України", постанова була прийнята на зборах акціонерів акціонерного товариства "ЕнегроАтом" наприкінці 2005 року щодо реорганізації компанії шляхом відокремлення виробництво та розподіл електроенергії. Компанія та магістральна мережа. У липні 2006 року реструктуризація компанії була завершена. В результаті філія ТУ "Енергозбут" - ДП "ЕнергоАтом" набуло нового статусу (при збереженні зареєстрованого місця в Києві).

Першочерговим завданням компанії є забезпечення споживачів правом на отримання надійної та безперебійної енергії для забезпечення необхідної кількості надійної та безперебійної енергії.

У рамках цього ДП "ЕнергоАтом" здійснює такі основні напрямки діяльності[7]:

• Покупка електроенергії на оптових та роздрібних ринках електроенергії;

• Продаж електроенергії споживачам (включаючи резидентів) на оптовому та роздрібному ринках електроенергії (потужності);

• Постачальники в по всій території України гарантують виконання функції.

інші види діяльності:

• Контроль за виконанням організаційно-технічних заходів щодо регулювання навантаження на електроенергію;

• Надавати послуги організаціям бухгалтерського обліку підприємств;

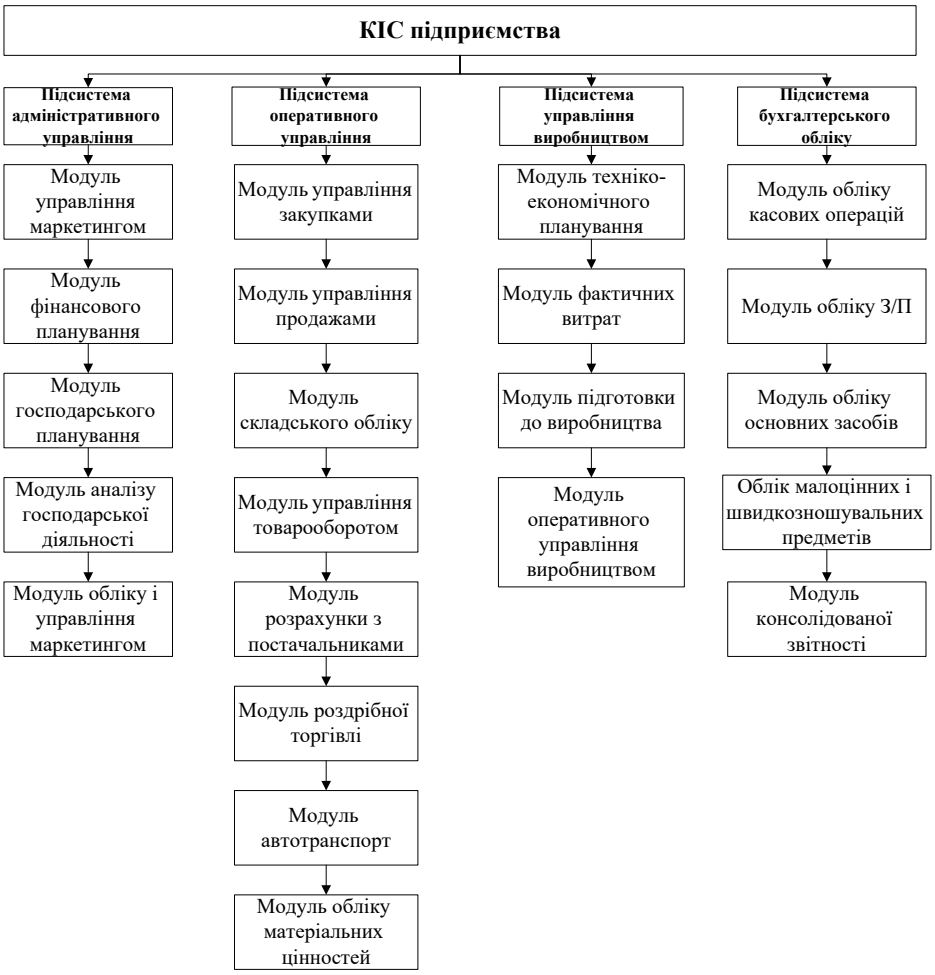

Рисунок 2 - Функціональна структура КІС досліджуваного підприємства

Першочерговим завданням компанії є забезпечення споживачів правом на отримання надійної та безперебійної енергії для забезпечення необхідної кількості надійної та безперебійної енергії.

У рамках цього ДП "ЕнергоАтом" здійснює такі основні напрямки діяльності[7]:

• Покупка електроенергії на оптових та роздрібних ринках електроенергії;

• Продаж електроенергії споживачам (включаючи резидентів) на оптовому та роздрібному ринках електроенергії (потужності);

• Постачальники по всій Україні гарантують виконання функції.

Інші види діяльності:

• Контроль за виконанням організаційно-технічних заходів щодо регулювання навантаження на електроенергію;

• Надавати послуги організаціям бухгалтерського обліку підприємств;

• Впроваджувати енергозберігаючі заходи;

• Дослідження стану та перспектив розвитку ринків електроенергії та потужностей.

Серед основних принципів стратегії розвитку компанії - підтримка позицій, що забезпечує постачальників по всій Україні ресурсами, застосування клієнтоорієнтованого підходу у всіх бізнес-процесах продажу енергії, поліпшення відносин із клієнтами та внутрішніх процесів. Компанія надає розширений спектр послуг.

1.2 Аналіз об'єкта автоматизації. Структура IT-відділу.

Питання підвищення ефективності роботи технічного відділу має бути вирішено до пікового періоду. Один із способів - запровадити систему виставлення рахунків за додатками.

Коли IT-відділ планують реорганізувати внаслідок, наприклад, розширення, збільшення робочого навантаження або плану реорганізації IT-відділу, існує нагальна потреба в системі показників для оцінки ефективності IT-відділу, тому необхідно зважати на це. Персонал приймає рішення та визначає IT-інфраструктуру галузі інвестицій.

Ось чому проблема створення системи баз даних служб для IT-відділу повинна бути вирішена задовго до пікового періоду. По-перше, можна уникнути багатьох проблем. По-друге, розробка ефективного інструменту моніторингу та аналізу є дуже серйозним завданням, що вимагає багато часу та інтелектуальних ресурсів, що неможливо уявити в надзвичайних ситуаціях.

Рисунок 3 - DFD – діаграма основних потоків даних на досліджуваному підприємстві

Наразі досвід у цьому виді роботи дуже важливий. Хочу відразу зазначити, що проект постійно розробляється, адже він впливає на багато сфер діяльності підприємства і може бути змінений або розширений під певні завдання.

Під час переддипломної практики я брав участь у розробці системи управління та аналізу послуг IT-відділу, після чого потрібно було написати модуль бази даних сервісів. Слід зазначити, що якщо рівень автоматизації низький, вимоги до IT-відділу (якщо такі є) можуть бути зменшені, щоб підтримувати ефективність роботи локальної мережі[2].

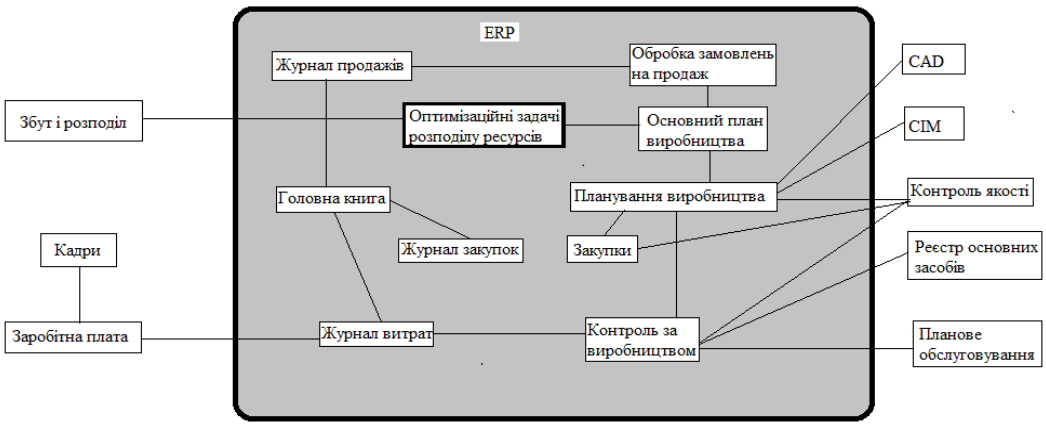

Рисунок 4 - Місце задачі оптимізації ресурсів підприємства серед інших задач

Метод розрахунку часової вартості деяких завдань такий: використовуємо систему бухгалтерських додатків IT-відділу, застосовуємо завдання до певної області, фіксуємо час подання заявки, початок і кінець програми та обчислюємо

його. Витрачений час (якщо працівників більше одного, множимо цей час на кількість людей, які беруть участь у процесі).

Завдяки аналізу можна визначити область завдання. Програмне забезпечення включає надання програмних консультаційних послуг, навчання працівників інших підрозділів програмного забезпечення тощо. Слід зазначити, що для подальшого детального та поглибленого аналізу вводиться класифікатор робочого циклу. Сюди необхідно входити один раз, щодня, щотижня, щомісяця або щокварталу.

Наприклад, виходячи із програмного забезпечення, частота змін програмного забезпечення та потреби у навчанні, певні робочі місця щодо програмного забезпечення, що використовують софт, віднесені до категорії «одноразово», тоді як інше програмне забезпечення віднесено до категорії «щомісяця». Потім ці класифікатори використовуються у шаблоні пам’ятки, отриманому IT-відділом, щоб швидко визначити пріоритет проблеми.

Формулюючи сферу відповідальності та весь проект, я врахував побажання відділу в цілому, а також рекомендації відділу щодо вибору середовища розробки та концепції побудови програмного забезпечення, продукту.

В результаті встановлення чіткого зворотного зв'язку та розподілу відповідальності покращилась якість контактів із партнерами з програмного та апаратного обслуговування, а також між IT-відділом та іншими департаментами.

2 ТЕХНІЧНЕ ЗАВДАННЯ

2.1 Найменування програми.

Найменування програми: База даних « сервісний центр »

2.2 Призначення і область застосування.

Розробити програмний комплекс, що складається з модулів «Збір даних» і «АРМ Техніка» з реалізацією WEB-інтерфейсу і форми даних, які вільно переносяться.

2.3 Вимоги до програми.

Необхідно вводити та реєструвати працівників підприємства, підтримувати базу даних усіх підрозділів та обмежувати права доступу відповідних підрозділів до інформації про працівників. Програмне забезпечення повинно бути легким для модифікації та розширення, щоб мати змогу підтримувати заповнений каталог даних і щоб адміністратор міг його в будь-який момент налаштувати[9].

Відповідно до вимог користувача, програмне забезпечення має генерувати такі документи у форматах Microsoft Word, MS XML, Open XML та PDF: «Запит на обслуговування», «Звіт про роботу принтера», «Звіт про роботу копіювальної машини», «Звіт про ПК», "Пов'язана загальна інформація про ПК". Подібним чином програма повинна дозволяти класифікувати апаратне та програмне забезпечення та здійснювати пошук відповідно до визначених критеріїв.

Основні вимоги до серверної частини:

- Надійність зберігання даних;
- Швидкий доступ до даних;
- Надмірність даних;
- Захищеність конфіденційних даних.

Основні вимоги до клієнтської частини:

- Незалежність від платформи;
- Інтуїтивно зрозумілий інтерфейс;
- Малі вимоги до апаратного забезпечення;

- Відповідність паперовим формам.

Працівники повинні здійснити набір організаційних та технічних заходів для забезпечення надійної (стійкої) роботи програми.

Список такий:

а) технічні засоби організації безперебійного живлення;

б) використання ліцензійного програмного забезпечення;

в) регулярно виконувати рекомендації, що містяться в постанові Міністерства праці та соціального розвитку Російської Федерації від 23 липня 1998 року про затвердження міжвідомчих стандартних специфікацій часу, що стосуються персональних послуг та офісного обладнання, а також програмного забезпечення ";

г) регулярно виконувати вимоги ГОСТ 51188-98. Захист інформації. Тестування комп'ютерного вірусу.

В умовах експлуатації апаратного та програмного забезпечення час відновлення (не фатальний збій операційної системи) після збою, спричиненого відмовою апаратного живлення (інші зовнішні фактори), не повинен перевищувати 30 хвилин.

Час відновлення після несправності, спричиненої апаратною помилкою або фатальною несправністю (збоєм), не повинен перевищувати часу, необхідного для усунення несправностей апаратного забезпечення та повторної інсталяції програмного забезпечення.

2.4 Умови експлуатації.

Кліматичні умови експлуатації, які повинні забезпечувати зазначені характеристики, повинні відповідати вимогам технічних засобів з точки зору їх експлуатаційних умов.

Мінімальна кількість персоналу, необхідного для роботи програми, повинна становити принаймні 2 штатних одиниці - адміністратор системи та кінцевий користувач програми - оператор. Системний адміністратор повинен

мати вищу освіту та сертифікат від виробника операційної системи. Список завдань, які виконує системний адміністратор, повинен включати[10]:

а) Завдання на підтримку працездатності технічних засобів;

б) встановлення системного програмного забезпечення та технічне обслуговування операційної системи;

До складу технічних засобів повинен входити IBM-сумісний персональний комп'ютер (ПЕОМ), що виконує роль сервера та включає в себе:

- Intel Pentium IV 1 MHz;
- RAM 128Мб;
- HDD 10Гб;
- Відео адаптер SVGA.

На фазі розробки технічного завдання слід виконувати фазу розробки, узгодження та затвердження технічного завдання.

На етапі проектування, слід реалізувати наступні етапи роботи:

- Розробка програми;
- Документація програми;
- Процедури тестування.

На етапі розробки технічних завдань слід виконати такі операції [3]:

1. Постановка проблеми
2. Визначення та тлумачення вимог до технічних засобів;
3. Визначення плану;
4. Визначення стадії розробки та умови плану;
5. Координація та затвердження технічного завдання.

Розробка документації програмного забезпечення повинна здійснюватися відповідно до вимог документації. На етапі тестування необхідно виконати такі види робіт:

1) Розробка, узгодження та затвердження кабінету та методів випробувань;

2) Налаштування програми та програмної документації відповідно до результатів випробувань.

3 АНАЛІЗ МОДЕЛІ ТА ПРЕДМЕТНОЇ ОБЛАСТІ

3.1 Опис предметної області.

Програмування - це синтез науки та мистецтва. Мета проекту - створити систему, яка[5]:

- Буде відповідати необхідним функціональним специфікаціям;

- Погоджуватися з обмеженнями, накладеними пристроєм;

- Виконати явні та неявні вимоги щодо продуктивності та споживання ресурсів;

- Виконувати явні та неявні стандарти проектування програмного забезпечення;

- Відповідати вимогам, спрямованості, тривалості, вартості та використанню інших інструментів самого процесу розробки.

Комп'ютерна програма - це набір основних інструкцій процесора, представлених у файлі у вигляді послідовності байтів (машинного коду). Кожна команда може кодуватися одним або кількома байтами. Цю форму програми можна скласти вручну, але через незручності безпосереднього управління процесором за допомогою простих команд люди просто не можуть дозволити собі таку роботу.

Тому програма написана як звичайний текст мовою програмування. Цей текст називається вихідним текстом (або вихідним кодом) програми.

Елементи дизайну програмного забезпечення можна розділити на три категорії:

1) Символи - мова, що використовується для написання кожної моделі;

2) Правила моделювання процесу проектування;

3) Інструмент, який прискорює процес створення моделі і відображає закон функціонування моделі.

Ці інструменти допомагають виявити помилки в процесі розробки.

Об'єктно-орієнтована модель відповідає на таке запитання: «Як найкраще розділити систему на підсистеми?» Це об'єктно-орієнтована декомпозиція.

Процес проектування базується на таких принципах: абстракція, інкапсуляція, модульність, ієрархічна структура, типізація, паралельність та стабільність.

Спрощений абстрактний опис, який вказує на деякі важливі атрибути, опускаючи інші важливі моменти, тобто описує характеристики та концептуальні межі, що відрізняють його від інших типів об’єктів. На цьому етапі опис мінімального зв’язку об’єкта та його поведінки не має побічних ефектів поза сферою його застосування.

Абстракція допомагає думати про те, як поводитися з програмами та внутрішніми пристроями (внутрішня структура об’єктів та реалізація їх методів). Насправді він складається з двох частин: інтерфейсу та реалізації.

Цей інтерфейс відображає зовнішню поведінку об'єкта.

Інкапсуляція - це процес відокремлення кожного елемента об'єкта, межі та поведінки його обладнання; ізоляція абстрактних договірних зобов'язань від їх реалізації[6].

Модульність - це атрибут системи, який був розділений на взаємопов’язані між собою модулі. Принципи абстракції, інкапсуляції та модульності доповнюють один одного. Об'єкти логічно визначають абстрактні межі, а інкапсуляція та модуляризація роблять їх фізично недоторканними.

3.2 Критичний опис існуючого програмного забезпечення.

В даний час багато компаній та сторонні розробників створили системи, подібні для мого проекту. Я переглянув такі пакети:

1) "Обладнання робочої станції та системний адміністратор", розроблений групою Smart-soft. Система громіздка, вимагає системних ресурсів і не містить усіх необхідних функцій, її неможливо розширити через відсутність модульної структури.

2) "Автоматизована система управління IT-інфраструктурою", розроблена Stels Technologies. Система може краще відображати потреби підприємств. Окрім ціни та крос-платформенності, система також спеціально розроблена та

реалізована для операційної системи Windows, тому її не можна використовувати на Linux або Mac OS X.

3) "Робоча станція системного адміністратора", розроблена компанією Small Soft Tech. Як і всі попередні системи, ця система розроблена для Windows і містить лише тісно пов'язану базу даних, яка не може задовольнити ваші потреби. Всі ці системи є сучасними та надійними, але вони не відповідають усім вимогам, я вирішив створити власні програмні продукти, які одночасно будуть масштабованими, кроссплатформенними та модульними. Ще однією відмінністю мого проекту є відносно низька вартість обслуговування та реконфігурації.

3.3 Об'єктна модель.

Об'єктно-орієнтована технологія базується на об'єктній моделі. Основними принципами є: абстракція, інкапсуляція, модульність, ієрархічна структура, типізація, паралельність та зміст. Нічого нового, але загалом - це вперше. Об'єктно-орієнтований аналіз та дизайн принципово відрізняється від конструктивного проектування. Програмування НУО - це метод організації програм у вигляді сукупності об’єктів. Кожен об'єкт є екземпляром певного класу, і клас утворює ієрархію успадкування [8].

Алгоритм - основним елементом є об'єкт:

• Рівень «стає частиною»;

• Будь-який об'єкт є екземпляром будь-якого класу;

• Заняття організовані в ієрархічній структурі

Програмування не ґрунтується на ієрархічних відносинах і не належить до ООП, а називається програмуванням на основі абстрактних типів даних.

ГО-дизайн - це метод програмування, який поєднує в собі процес декомпозиції об'єкта і технологію представлення логічних і фізичних методів, а також статистичні та динамічні моделі спроектованої системи.

Відмінність від структурного проектування: декомпозиція представлена у вигляді класів і об'єктів та у вигляді алгоритмів в спільних підприємствах.

Об'єктно-орієнтований аналіз - це метод, при якому потреби системи сприймаються відповідно до класів і об'єктів, які ідентифіковані в предметній області.

ООА - формує об'єкт, на якому базується ООП, який є основою для реалізації в ООП.

3.4 Аналіз обраного середовища розробки та мови програмування.

Основна функція мови C# полягає в тому, що вона зосереджена на платформі Microsoft .NET. Творці C# намагалися надати розробникам природний спосіб отримати доступ до всіх функцій платформи .NET. Очевидно, що це рішення є більш-менш обов'язковим, оскільки платформа .NET спочатку надавала набагато більше можливостей, ніж будь-яка мова програмування (C ++), що використовувалася на той час, маючи при цьому простоту Visual Basic.

Крім того, творці C# хотіли приховати від розробників якомога більше дрібних технічних деталей, включаючи типи пакування / розпакування, ініціалізацію змінних та збір сміття. Це дозволяє програмістам C# краще зосередитись на змісті завдання. Вирішуючи цю проблему, розробники C# намагалися врахувати уроки, отримані в результаті реалізації Visual Basic.

Порівняно з розширенням існуючих мов, ще однією перевагою створення нових мов програмування є те, що при створенні нових мов немає потреби турбуватися про зворотну сумісність, що часто вкрай ускладнює вирішення старих проблем або навіть введення нових атрибутів у мовні стандарти.

C# знаходиться в середньому положенні: найбільш надокучливі та неоднозначні функції C ++ були вилучені зі стандарту мови, але в той же час мова зберігає потужну силу, властиву таким мовам, як C ++, Java або VB.

Рік випуску Visual Studio 2008 - це подальший розвиток середовища розробки додатків, широко використовуваного в .NET Framework попереднього покоління (Microsoft Visual Studio 2005). Нове середовище розробки дозволяє створювати та використовувати служби програм, створювати веб-сайти ASP.NET з додатковими функціями, створювати привабливі настільні програми

з розширеними функціями, розробляти описи бізнес-процесів, створювати служби та вдосконалювати програми Microsoft Office.

Visual Studio 2008 включає набір нових утиліт, корисних удосконалень, нового дизайнера та візуального редактора. Головною метою всіх змін є скорочення часу розробки додатків, що використовують новітні технології, що входять до .NET Framework 3.0 та 3.5.

Основні зміни в мовах програмування C # та VB.NET

• LINK (Мовний інтегрований запит) - мова, що використовується для запитів різних структур даних, одну з яких можна вибрати окремо.

o Запит об'єкта

o Запит до бази даних

o запит XML

o Запит власних типів даних

• Створення LINK вимагає зміни мови програмування та введення нових синтаксичних структур.

o процедура ініціалізації об'єкта;

o методи розширення;

o лямбда-вираз;

Методи розширення, як метод додавання нових елементів до існуючих класів без перекомпіляції;

• Основа функціонального програмування - лямбда-оператори та лямбда-вирази як новий спосіб оголошення делегатів;

• Інтегрування технології AJAX із .NET Framework без необхідності встановлювати додаткові зовнішні бібліотеки;

o Вбудовані шаблони для створення веб-сайтів AJAX ASP.NET;

o Нові засоби управління, які вирішують основні технічні проблеми при використанні AJAX;

o Можливість підключення ASP.NET AJAX Toolkit, як стандартний інструмент при створенні веб-додатків;

• Дозвіл на використання нових засобів управління технологією LINK;

• Новий візуальний редактор та режим його роботи;

• Додано розумні функції та налагоджено під час використання JavaScript;

• Windows Presentation Foundation (WPF) - нова технологія для створення настільних додатків Windows з багатими графічними інтерфейсами;

о Використання двовимірної та тривимірної графіки, мультимедіа, включаючи анімацію та перетворення управління, використання векторної графіки та функцій DirectX;

о Нова номенклатура для контролю будівельних додатків;

о Декларативна мова для опису структури об'єктів-XAML, як новий метод опису графічних інтерфейсів та їх характеристик

Візуальний редактор та редактор XAML для створення ефективних додатків;

• Windows Workflow Foundation (WF) - нова технологія, що використовується для опису процесу реакції;

о побудова опису бізнес-процесу (робочого процесу) та інтегрування різних типів процесів в один опис;

о Візуальний редактор для встановлення опису процесу;

о вбудований стандартний набір управління для опису процесу;

• Windows Communication Foundation (WCF) - нова технологія побудови та використання сервісів, заснована на концепції сервісно-орієнтованої архітектури (SOA);

о Створений та використаний опис комбінації;

 Веб-сервіс;

Remote - віддалена обробка .NET;

 Покращення веб-сервісу (WSE);

о Використання стандартів WS- *;

4 СТРУКТУРА ТА АЛГОРИТМІЗАЦІЯ БАЗИ ДАНИХ

Моє завдання - створити зручну програму для підприємства, за допомогою якої різні відділи можуть отримувати інформацію в зручному для них вигляді і складати заявки на рішення тих чи інших завдань:

• отримувати інформацію про апаратне та програмне забезпечення в короткій або повній формі;

• Вводити нову інформацію про нове обладнання, нове програмне забезпечення і т. д.

• Редагувати існуючу інформацію, таку як зміна інвентарного номера, місця розташування, встановленого програмного забезпечення і т. д.

• Отримувати вихідні документи в певному форматі, наприклад, додатки для ремонту і заправки друкувального обладнання, поведінка при скасуванні реєстрації обладнання, поведінка при установці програмного забезпечення і стан комп'ютерів в мережі.

Оскільки база даних, в якій зберігається вся інформація, може розташовуватися як в локальній, так і в глобальній мережі, наприклад, вона може розташовуватися на сайті, тому під час установки і налаштуванні клієнтської програми системний адміністратор може виконувати наступні операції: він вказує сервер бази даних і аутентифікаційну інформацію в ньому.

В цілому написана програма задовольняє всім вимогам, що пред'являються і зручна у використанні завдяки інтуїтивно зрозумілому інтерфейсу.

4.1 Зразки бланків вихідних документів.

Вихідними документами є різні технічні паспорти, звіти про встановлене програмне забезпечення, звіти про техобслуговування і заправку друкувального обладнання, запити на технічне обслуговування і заправку, а також докази списання обладнання. Всі вони відповідають національним стандартам ведення документів про обіг бізнесу і статутом.

4.2 Вибір інструментального середовища розробки.

Наступні критерії вибору здаються важливими для визначення середовища розробки програмного забезпечення:

Операційна оболонка програмного комплексу буде ОС Windows;

- Робота комплексної будівлі повинна бути максимально автоматизована, щоб надати користувачам максимально зручний інтерфейс моніторингу та налаштування;

- Середовище приладу повинно мати розширені засоби налагодження;

- Повинен надавати потужні й гнучкі інструменти часу розробки (такі як стандартні бібліотеки компонентів, браузери об'єктів, браузери баз даних і т. д.)

Виходячи з цих критеріїв, було вирішено використовувати наступне седовище розробки для створення програмного комплексу:

- Visual C # робить розробку потужних додатків Windows швидким процесом. Тепер одна людина може використовувати Visual C # для написання трудомістких додатків Windows, таких як Pascal або чистий C.

- Microsoft SQL Server 2008. Сервер бази даних забезпечує можливість одночасної роботи декількох користувачів з одним інформаційним простором, і система не вимагає постійного спостереження з боку системного адміністратора. Він надає дуже зручний і ефективний інтерфейс для доступу до бази даних.

4.3 Постановка завдання.

Перед написанням дипломної роботи я: вивчив можливості Microsoft VisualStudio, зібрав матеріали для написання дипломної роботи, а потім приступив до розробки дипломного проекту. Створена мною база даних містить всю інформацію про пристрої для друку (принтери, копіювальні апарати, багатофункціональні пристрої), програмномне забезпечення і кластери персональних комп'ютерів.

Microsoft VisualStudio випуску 2008 - це інтегроване середовище розробки для додатків Windows, що включає наступні компоненти:

• Високопродуктивний компілятор від мови високого рівня до машинного коду;

• Об'єктно-орієнтовані модулі і компоненти;

• Візуальні додатки з прототипів;

• Інструменти для побудови баз даних.

Вбудований компілятор Microsoft VisualStudio забезпечує високу продуктивність, необхідну для створення клієнт-серверних додатків. Компілятор на даний момент найшвидший в світі (неофіційні дані). Він забезпечує простий процес розробки блоку програмування на мові четвертого покоління (4GL) і скорочує час виконання робіт, а також забезпечує унікальну якість коду компілятора 3GL.

Оскільки Microsoft VisualStudio автоматично готує необхідні шаблони програм і відповідні файли ресурсів, візуальний дизайн форми полегшує програмістам розробку багатьох аспектів інтерфейсу програми. Бібліотека візуальних компонентів надає програмістам різні шаблони програмного забезпечення, створені розробниками Microsoft VisualStudio. Ці шаблони можна використовувати відразу ж або використовувати в програмі після простих налаштувань.

4.4 Загальні відомості про програмне забезпечення.

Ядром пакета є база даних, розташована на сервері Microsoft SQL Server 2008 Enterprise Edition. База даних складається з 5 таблиць, що містять різні дані, включаючи загальну інформацію про структуру підприємства і дані, безпосередньо пов'язані з апаратним і програмним забезпеченням. Нижче наводиться структура таблиці бази даних.

Ці таблиці відображають повну структуру бази даних, задіяної в проекті, і пояснюють значення всіх полів в базі даних. Програмісти, які хочуть додати або розширити системні функції, можуть використовувати цю інформацію.

Цей комплекс побудований за модульною конструкції, і більшість модулів мають схожі типи, тому основні модулі будуть описані нижче.

Перший модуль, який бачить користувач, - це система, яка відображає повну інформацію про доступні друкуючі пристрої і їх повні технічні описи, коли це необхідно.

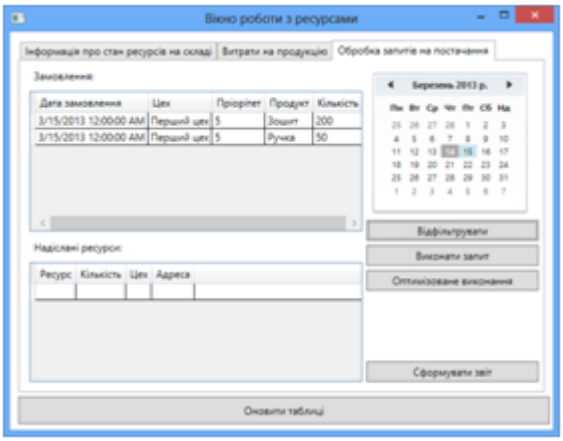

Рисунок 5 – Відображення доступних пристроїв у додатку

Частина програмного коду:

<сітка>

<Grid.RowDefinitions>

<RowDefinition Height = "Auto" MinHeight = "21" />

<Визначення рядка />

<RowDefinition Height = "Auto" MinHeight = "22" />

</Grid.RowDefinitions>

<Меню Grid.Row = "0" Horizo ntalAlignment = "Stretch">

<MenuItem Header = "Файл">

<MenuItem Header = "Імпорт XML" />

<Розділювач />

</menu item>

<MenuItem Header = "Зміст">

<MenuItem Header = "Corpus" Click = "S_Korpus_Click" />

<MenuItem Header = "Підрозділи" Клацніть = "S_Depart_Click" />

<MenuItem Header = "Технологія друку" Click = "S_Tech_Click" />

</ пункт меню>

<MenuItem Header = "Завдання">

<MenuItem Header = "Account" Клацніть = "Account" />

<MenuItem Header = "Repair" />

<MenuItem Header = "Move" click = "Move_Click" />

</пунктменю>

<MenuItem Header = "Звіти">

<MenuItem Header = "Repair" />

<MenuItem Header = "Mobile" />

<MenuItem Header = "Підсумковий звіт" />

</пунктменю>

<MenuItem Header = "Help">

<MenuItem Header = "Match..." />

<MenuItem Header = "Про ..." />

</пунктменю>

</Меню>

String All_Record = "SELECT T_Department.name, T_Printer.inventar, T_Prn_Modern.p_id, T_Prn_Modern.m_type, T_Prn_Modern.m_date_z," +

"T_Prn_Modern.id, T_S_Printer.p_firm, T_S_Printer.p_model" +

"FROM T_Prn_Modern INNER JOIN" +

"T_Printer ON T_Prn_Modern.p_id = T_Printer.id ВНУТРІШНЄ ПРИЄДНАННЯ" +

"T_S_Printer ON T_Printer.p_id = T_S_Printer.id ВНУТРІШНЄ ПРИЄДНАННЯ" +

"T_Department ON T_Printer.d_id = T_Department.id WHERE T_Prn_Modern.m_made ='False' AND T_Printer.active ='True'";

Example SqlDataAdapter;

Підключення SqlConnection;

Приватна порожнеча UpdateView ()

{

КомандаSqlCommand =новийSqlCommand();

comd.CommandText = All_Record;

comd.Connection = conn;

адаптер. command=command;

Набір даних набору даних = новий набір даних ();

адаптер.Заповнення (набір даних, "T_Prn_Modern");

m_orders = набір даних. Таблиці ["T_Prn_Modern"];

DGC_Main.ItemsSource = m_orders.DefaultView;

}

...

Наступні модулі використовуються для заповнення довідкових даних та форм бухгалтерського обліку. Ці модулі дозволяють взаємодіяти з базою даних у зручній формі та складати звіти, необхідні для функціонування системи.

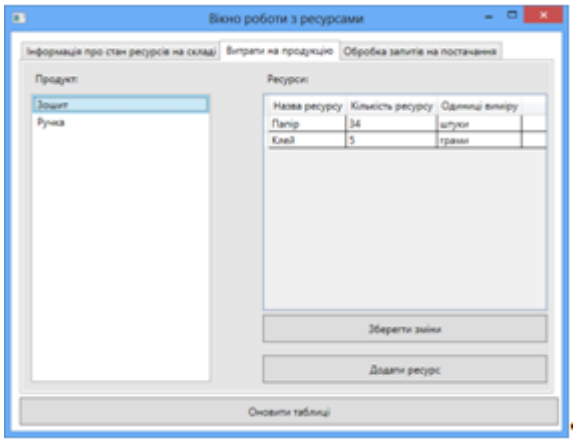

Рисунок 6 – Форма з відображенням результатів діяльності для складання звітів

Частина програмного коду:

...

<Вікно x: Клас = "ASU_Tech.W_Depart"

xmlns: local = "clr-space name: ASU_Tech"

Title = "каталог відділів" height = "266" width = "644">

<window.resource>

<СтильTargetType = "{x: Type TextBlock}">

<Setter property = "FontFamily" value = "Times New Roman" />

</style>

<local: StKorpNameConvertor x: Key = "StKorpNameConvertor"> </local: StKorpNameConvertor>

</ Window.Resources>

<сітка>

<Grid.ColumnDefinitions>

<ColumnDefinition Width = "" />*

<ColumnDefinition Width = "Автоматичний" MinWidth = "3" />

*<ColumnDefinition Width = "2 *" />*

</Grid.ColumnDefinitions>

<GridSplitter Grid.Column = "1" ResizeBehavior = "PreviousAndNext" ResizeDirection = "Columns" Width = "3" VerticalAlignment = "Stretch" />

<ListBox Margin="3,3,3,3" Name="LB_Depart" DisplayMemberPath="Name" SelectionChanged="LB_Dep_Changed" />

<Grid Grid.Column = "2" DataContext = "{Binding ElementName = LB_Depart, Path = SelectedItem}">

<Grid.RowDefinitions>

<RowDefinition Height = "Auto" />

<RowDefinition Height = "Автоматичний" />

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width = "Auto" />

<Визначення стовпця />

</Grid.ColumnDefinitions>

<TextBlock Text = "Title" Margin = "3" Grid.Column = "0" Grid.Row = "0" Horizo ntalAlignment = "Center" VerticalAlignment = "Center" />

<TextBlock Text = "Type" Margin = "3" Grid.Column = "0" Grid.Row = "1" HorizontalAlignment = "Center" VerticalAlignment = "Center" />

<TextBlock Text = "Audience" Margin = "3" Grid.Column = "0" Grid.Row = "3" Horizo ntalAlignment = "Center" VerticalAlignment = "Center" />

<TextBlock Text = "Head" Margin = "3" Grid.Column = "0" Grid.Row = "4" HorizontalAlignment = "Center" VerticalAlignment = "Center" />

<TextBlock Text = "Внутрішній телефон." Margin = "3" Grid.Column =" 0" Grid.Row = "5" Horizo ntalAlignment =" Center" VerticalAlignment = "Center" />

<TextBlock Text = "Зовнішнійтелефон". Margin = "3" Grid.Column = "0" Grid.Row = "6" Horizo ntalAlignment = "Center" VerticalAlignment = "Center" />

<TextBox name="TB_Name" Grid.Column="1" Grid.Row="0" margin="3" text="{Шляхприв'язки=Ім'я}" />

<ComboBox name="CB_Type" Grid.Column="1" Grid.Row="1" margin="3" text="{binding path=D_type}">

</полезісписком>

<ComboBox Name = "CB_Korpus" Grid.Column = "1" Grid.Row = "2" Margin = "3" Text = "{Шлях прив'язки = Korp_name, Converter = {StaticResource StKorpNameConvertor}}">

<ComboBox.ItemTemplate>

<Шаблон даних>

<TextBlock Text = "{Binding Path = Number, Converter = {StaticResource KorpNameConvertor}}"> </TextBlock>

</ Шаблон даних>

</ComboBox.ItemTemplate>

</ поле зі списком>

<TextBox Name = "TB_Room" Grid.Column = "1" Grid.Row = "3" Margin = "3" Text = "{Binding Path = Room}" />

<TextBox Name = "TB_Mainhead" Grid.Column = "1" Grid.Row = "4" Margin = "3" Text = "{Binding Path = Mainhead}" />

<TextBox name = "TB_TelIn" Grid.Column = "1" Grid.Row = "5" margin = "3" text = "{binding path = Tel_in}" />

<TextBox Name = "TB_TelOut" Grid.Column = "1" Grid.Row = "6" Margin = "3" Text = "{Шлях прив'язки = Tel_out}" />

</grid>

</window>

...

Наступний модуль дозволяє зареєструвати новий друкарський пристрій. Модуль використовує автоматичне заповнення, тому після заповнення деяких полів за раз користувач може використовувати ці дані, вводячи лише кілька перших літер.

Рисунок 7 – Форма додавання нового пристрою в систему

Частина програмного коду:

...

<window.resource>

<СтильTargetType = "{x: Type TextBlock}">

<Setter Property = "FontWeight" value = "Bold" />

<Setter property = "FontFamily" value = "Times New Roman" />

</style>

<local: PrnNameConverter x: key = "PrnNameConverter"> </ local: PrnNameConverter>

</ Window.Resources>

<сітка>

<Grid.RowDefinitions>

<RowDefinition Height = "Auto" MinHeight = "28" />

<RowDefinition Height = "Auto" MinHeight = "28" />

<RowDefinition Height = "Auto" MinHeight = "28" />

<RowDefinition Height = "Auto" MinHeight = "28" />

<RowDefinition Height = "Auto" MinHeight = "22" />

<RowDefinition Height = "Auto" MinHeight = "28" />

<визначення висоти рядка = "" />*

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width = "Авто" MinWidth = "85" />

<ColumnDefinition Width = "" />*

</Grid.ColumnDefinitions>

<TextBlock Grid.Row = "0" Grid.Column = "0" Text = "Department" Margin = "3" Horizo ntalAlignment = "Center" />

<TextBlock Grid.Row = "1" Grid.Column = "0" Text = "Printer" Margin = "3" Horizo ntalAlignment = "Center" />

<TextBlock Grid.Row = "2" Grid.Column = "0" Text = "Inv. Number" Margin = "3" Horizo ntalAlignment = "Center" />

<TextBlock Grid.Row = "3" Grid.Column = "0" Text = "Vendor" Margin = "3" Horizo ntalAlignment = "Center" />

<TextBlock Grid.Row = "4" Grid.Column = "0" Text = "Дата доставки" Margin = "3" Horizo ntalAlignment = "Center" />

<TextBlock Grid.Row = "5" Grid.Column = "0" Text = "Warranty" Margin = "3" Horizo ntalAlignment = "Center" />

<Button Margin = "3" Grid.Row = "6" Width = "Auto" Height = "Auto" Horizon ntalAlignment = "Left" MinWidth = "75" MaxHeight = "30" Click = "Cancle_Click"> Скасувати < / Кнопка>

<Button Margin = "3" Grid.Row = "6" Grid.Column = "1" Width = "Auto" Height = "Auto" Horizo ntalAlignment = "Right" MinWidth = "75" MaxHeight = "30" Click = "Accept_Click"> Прийняти </ button>

<ComboBox Margin = "3" Grid.Row = "0" Grid.Column = "1" Name = "CB_Depart" DisplayMemberPath = "Name" />

<ComboBox Margin = "3" Grid.Row = "1" Grid.Column = "1" Name = "CB_Prn">

<ComboBox.ItemTemplate>

<Шаблон даних>

<TextBlock DataContext = "{Binding}">

</ текстовий блок>

</ Шаблон даних>

</ComboBox.ItemTemplate>

</ поле зі списком>

<ComboBox margin = "3" Grid.Row = "3" Grid.Column = "1" name = "CB_Firm" IsEditable = "True" />

<TextBox margin = "3" Grid.Row = "2" Grid.Column = "1" name = "TB_Inventar" />

<xcdg: DatePicker Grid.Column="1" Grid.Row="4" Margin="3" Name="datePicker1" xmlns:xcdg="http://schemas.xceed.com/wpf/xaml/datagrid" / >

<xcdg: MaskedTextBox Mask = "00" Grid.Column =" 1" Grid.Row = "5" Margin =" 3,3,3,3" Name = "TB_Warantary" xmlns: xcdg = "http: // mode. xceed.com/wpf/xaml/datagrid />

</ grid>

</ window>

...

Цей модуль дає поняття про структуру та функції, що використовуються в системі розробки. Наступний модуль - це проект, який дозволяє віддалено отримувати різні дані та експортувати отримані дані у зручній формі за допомогою XML та XLST, завдяки чому ви можете передавати отримані дані в інші програми для аналізу та обробки. Подальша обробка.

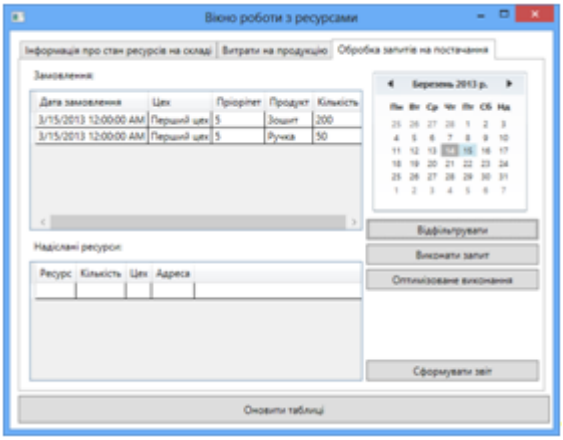

Рисунок 8 – Форма для віддаленого отримання даних та передача їх у сторонні програми для подальшої обробки

Частина програмного коду:

...

private void GetProductInfo (сфера управлінняScope)

{

Запит ObjectQuery = новий ObjectQuery («ВЕРЕСТИ ВЕРСІЮ, InstallDate, ім'я ВІД Win32_Product»);

Пошуковий засіб ManagementObjectSearcher = новий ManagementObjectSearcher (обсяг, запит);

ManagementObjectCollection queryCollection = searcher.Get();

LV_Data.Items.Clear();

LV_Data.Columns.Clear();

LV_Data.Columns.Add("Ім'я");

LV_Data.Columns[0]. Ширина = 280;

LV_Data.Columns.Add ("Дата встановлення");

LV_Data.Columns [1]. Ширина = 90;

LV_Data.Columns.Add("Версія");

LV_Data.Columns [2]. Ширина = 85;

foreach (запит об'єкта управління m у колекції)

{

ListViewItem litem = новий ListViewItem(m ["ім'я"].ToString());

litem.SubItems.Add(String.Format("{0}.{1}.{2}", m["InstallDate"].ToString().Підрядок(0,4), m["InstallDate"].ToString () .Substring(4, 2), m ["InstallDate"].ToString().substring(6, 2)));

litem.SubItems.Add(m["Версія"].ToString());

LV_Data.Items.Add (items);

litem = (ListViewItem) litem.Clone ();

litem.SubItems.Clear ();

}

```
}
private void GetMainInfo(ManagementScope scope)
{
Запит ObjectQuery = новий ObjectQuery ("ВИБРАТИ * З ВІД Win32_OperatingSystem");
Пошуковий засіб ManagementObjectSearcher = новий ManagementObjectSearcher (обсяг, запит);
ManagementObjectCollection queryCollection = searcher.Get();
LV_Data.Items.Clear();
LV_Data.Columns.Clear();
LV_Data.Columns.Add("Назва ПК");
LV_Data.Columns[0]. Ширина = 50;
LV_Data.Columns.Add ("Каталог Windows");
LV_Data.Columns [1]. == 100;
LV_Data.Columns.Add("Ім'я");
LV_Data.Columns [2]. Ширина = 155;
LV_Data.Columns.Add("Версія");
LV_Data.Columns [3]. Ширина = 85;
LV_Data.Columns.Add ("Виробник");
LV_Data.Columns [4]. Ширина = 135;
foreach (запит об'єкта управління m у колекції)
{
ListViewItem litem = новий ListViewItem(m["csname"].ToString());
litem.SubItems.Add(m["WindowsDirectory"].ToString());
litem.SubItems.Add(m["Заголовок"].ToString());
litem.SubItems.Add(m["Версія"].ToString());
litem.SubItems.Add(m ["Виробник"].ToString());
LV_Data.Items.Add(item);
}
```

}

...

Усі ці модулі дають повну картину проекту, що розробляється, а також дозволяють простежити загальну структуру взаємозв'язку. При розробці цих модулів були використані сучасні технології та компоненти, а потім я коротко ознайомлю з основними технологіями, які допомогли створити програмне забезпечення.

4.5 Захист програмного продукту.

Будь-який розробник, який займається розробкою комерційних додатків, рано чи пізно зіткнеться з проблемою організації захисту додатків від копіювання та хакерських атак. Зазвичай власна розробка якісних систем захисту - це досить складний і трудомісткий процес. Тому більшість розробників намагаються вибрати більш просте і швидке рішення з використанням готових модулів системи захисту.

У більшості випадків захист заснований на використанні файлів ліцензій або серійних номерів. Надайте користувачам пробну версію програми, яка містить безліч обмежень. Наприклад, програмне забезпечення, яке я розробив в пробній версії, має обмежені можливості друку і час роботи. Користувач намагається запустити програму, і після визначення того, що програма підходить для нього, і оплати мита, файл ліцензії буде відправлений користувачеві. При наявності такого файлу програма починає працювати з повним функціоналом. Безкоштовне розповсюдження файлів ліцензій обмежене. У моєму проекті файли створюються з прив'язкою до обладнання комп'ютера користувача, тому їх не можна використовувати на інших комп'ютерах. У той же час система безпеки використовує шифрування даних, щоб зловмисникам було складніше зламати програму і відключити прив'язку до файлу ліцензії.

Реалізація такої програми ліцензування - непросте завдання, яке само по собі має безліч нюансів. Однак захист моїх програмних продуктів, написаних для Microsoft.Net, цим не обмежується. Крім посилання на файл ліцензії і

запобігання копіювання, вихідний код програми також повинен бути захищений від перегляду. Якщо додаток ніяким чином не захищений, ви можете використовувати спеціальні інструменти, такі як .NET Reflector, для декомпіляції протягом декількох хвилин і відновлення вихідного коду програми в формі, придатної для дослідження.

Програмний продукт .Net Reactor від Eziriz e.K. містить всі інструменти, необхідні для захисту мережевих додатків. Реалізована в ньому технологія NecroBit захищає додатки від декомпіляції шляхом змішування керованого коду і машинного (нативного) коду. .Net Reactor включає методи обфускації класів, типів і змінних, доповнені шифруванням рядків, використанням недрукованих символів при обфускації імен та навіть можливість обфускації блок-схем управління. У цьому продукті реалізована розширена схема ліцензування додатків. І, що найголовніше, .Net Reactor набагато дешевше більшості альтернативних методів захисту, які декларуються розробниками прямо на сторінках своїх сайтів.

Щоб забезпечити захист в моїй програмі, я використовував зазначену вище програму для генерації деяких класів, що дозволяє мені захистити додаток, не порушуючи його цілісності. Інтерфейс програми зручний і інтуїтивно зрозумілий.

На першому етапі вам потрібно вказати режим захисту, вибрати основну захищену збірку і необов'язкові додаткові збірки, а також вказати, чи хочете ви використовувати захист від декомпіляції і обфускації NecroBit.

Net Reactor підтримує два режими захисту: захист бібліотеки і захист додатків. Якщо обраний режим захисту програми, інші збірки і основна збірка будуть об'єднані в один файл. У режимі захисту бібліотеки інші збірки можуть бути захищені індивідуально або разом з основною збіркою. Крім того, захист в режимі бібліотеки захищає збірку, щоб її можна було використовувати з інших додатках в майбутньому.

На другому етапі параметри захисту були відрегульовані більш точно,

зокрема, включаючи такі функції:

• Включення стиснення ассемблерного коду;

• Включення підтримки компактної платформи;

• Налаштування параметрів плутанини.

Для забезпечення захисту встановлюється будь-яка комбінація обмежень, і вказується, що в цьому випадку демонстраційна версія повинна зупинятися тільки після спрацьовування всіх зазначених обмежень. При запуску програми nagscreen режим відображення також буде включений і вказувати день, з якого буде відображатися демонстраційна версія.

Третій крок - налаштувати диспетчер ліцензій. Це обмеження для конкретного додатка, яке буде застосовуватися при наявності файлу ліцензії. Тобто, якщо файл ліцензії відсутній, використовується набір обмежень, вказаний на другому кроці. Якщо він існує, вкажіть набір на третьому кроці. Перемикання між одним набором обмежень і іншим виконується автоматично. Отже, для того, щоб демонстраційна версія була повністю функціональна, користувачеві досить помістити файл ліцензії в каталог додатки.

На третьому кроці набору обмежень є додаткове обмеження апаратного блокування. Він використовується для створення файлу ліцензії, прив'язаного до певного комп'ютера. Прив'язка виконується на основі будь-якої комбінації ідентифікаторів обладнання комп'ютера (материнська плата, процесор, жорсткий диск, мережева карта).

Останній крок - власне згенерувати рівень захисту. Весь процес повністю автоматизований і займає кілька секунд. .Net Reactor створює підкаталог в каталозі, де знаходиться захищена основна збірка, і поміщає в нього всі файли і сценарії конфігурації, необхідні для збірки, потім включає їх до проекту і може використовуватися для компіляції всіх додатків або їхніх модулів.

4.6 Архітектура віддаленого опитування і занесення в БД.

Проаналізувавши і обравши найкраще рішення, я застосував у проекті технологію WMI.

Архітектура технології WMI включає наступні аспекти:

• Інфраструктура управління, включаючи диспетчер об'єктів CIM, який забезпечує стандартизований доступ до даних управління для додатків і центральний репозиторій даних управління даними, який називається репозиторієм диспетчера об'єктів CIM.

• Провайдери WMI - є посередником між диспетчером об'єктів CIM і керованим об'єктом. Провайдер використовує WMI API для передачі даних від керованого об'єкта диспетчеру об'єктів CIM, обробки запиту від імені керівника програми та створення повідомлень про події.

Інфраструктура управління складається з диспетчера об'єктів CIM і сховища диспетчера об'єктів CIM. Додаток покладається на диспетчер об'єктів для обробки інтерфейсу між керуючим додатком і постачальником даних. WMI полегшує цей зв'язок, надаючи загальний програмний інтерфейс для служб управління Windows за допомогою COM. COM API надає службі повідомлення про події та обробку запитів і може використовуватися в декількох середовищах програмування (таких як C і C ++). Репозиторій диспетчера об'єктів CIM містить архітектуру і розширення CIM, а також дані та інформацію про джерела даних. При обробці запитів на керовані об'єкти від керуючих додатків диспетчер об'єктів CIM буде використовувати дані схеми в цьому репозиторії.

Постачальник WMI - це стандартний сервер COM і DCOM, який виступає в якості посередника між керованим об'єктом і диспетчером об'єктів CIM. Якщо диспетчер об'єктів CIM отримує запит даних від керуючого додатку, якого немає в репозиторії диспетчера об'єктів CIM, або повідомлення про подію, яке може не підтримуватися диспетчером об'єктів CIM, диспетчер перенаправляє запит постачальнику WMI.

Для реалізації провайдера необхідно використовувати один з наступних

підтримуваних типів серверів:

• Служба Microsoft Windows 2008, локальна або віддалена.

• Локальний або віддалений стандартний виконуваний файл (EXE).

• Вбудована бібліотека динамічного компонування (DLL).

Зверніть увагу, що рекомендується використовувати локальні або видалені служби Windows і стандартні виконувані файли.

WMI поставляється з вбудованими постачальниками (або стандартними постачальниками), які надають дані з таких джерел, як реєстр.

WMI також забезпечує підтримку сторонніх постачальників. Ці постачальники можуть використовуватися для обробки запитів керованих об'єктів в певному середовищі. Зазвичай постачальники використовують MOF для визначення і створення класів. Постачальник використовує WMI API для доступу до сховища WMI і відповіді на запити диспетчера об'єктів CIM, відправлені додатком.

Вартість ПО складає - 95 678 грн.

З урахуванням даних про вартість комплекту програми, вартості установки і часткової вартості розробки, відсоток прибутку від однієї установки може скласти (для даної розробки) 10.5%.

Якщо прийняти ставку податку на додану вартість в 20% і з огляду на вартість програми, відсоток прибутку від установки і ставку податку на додану вартість, сума прибутку від кожної установки може скласти 4 000 грн.

ВИСНОВКИ

1. У даній випускній роботі - розглянуто корпоративний пакет програм, що містить інформацію про апаратне та програмне забезпечення.

2. Згідно з технічним завданням оцінена економічна ефективність розробленого програмного комплексу.

3. Розглянуто існуючі методи програмної реалізації засобів автоматизації, що використовуються для збору, обробки та зберігання різних параметрів ПК і облікової інформації для автоматизації роботи технічних фахівців. Була обрана форма реалізації системи - були визначені операційний інформаційний центр (ОІВ), його функції та структура, а також функції, які підтримуються інформацією про реалізацію.

4. Розроблено підсистему обміну інформацією між двома клієнтськими додатками розподільної бази даних.

5. Створено базу даних.

Це забезпечує:

• зберігання інформації про апаратне та програмне забезпечення;

• Зручний інтерфейс користувача.

6. Для підприємства створена зручна програма, за допомогою якої відділ інформаційних технологій може:

• Отримувати інформацію про доступне обладнання і вимоги щодо його ремонту і технічного обслуговування;

• Виконувати віддалений моніторинг програмного і апаратного забезпечення на ПК корпоративних користувачів.

7. Розраховані економічні вигоди від впровадження програмного продукту.

8. Таким чином, всі зазначені вимоги повністю виконані. Програмний комплекс працює нормально.

СПИСОК ЛІТЕРАТУРИ

1. Ніколаєнко К.І. Корпоративні інформаційні системи: Навчальний посібник / К.І. Ніколаєнко. – К. : КНЕУ, 2014. – 291 с.
2. ERP – системи [Електронний ресурс] / матеріал з мережі – режим доступу до ресурсу: https://trinion.org/articles/chto-takoe-erp-sistema
3. SAP ERP [Електронний ресурс] / матеріал з мережі – режим доступу до ресурсу: http://asapcg.com/press-center/articles/sap-r3
4. Microsoft Dynamics [Електронний ресурс] / матеріал з мережі – режим доступу до ресурсу: https://dynamics.microsoft.com/ru-ru/
5. Oracle E-Business Suite [Електронний ресурс] / матеріал з мережі – режим доступу до ресурсу: https://ru.wikipedia.org/wiki/Oracle_E-Business_Suite
6. Архітектура клієнт-сервер [Електронний ресурс] / дизайн систем – режим доступу до ресурсу: https://sergeygavaga.gitbooks.io/kurs-lektsii-testirovanie-programnogo-obespecheni/content/lektsiya-6-ch1-arhitektura-klient-server.html
7. Мінченко В.Г., Попов К.В. Інформаційні системи та технології: Навчальний посібник / В.Г. Попов. – К.: МАУП, 2013. – 192 с.
8. Нікіфоров Т.Р., Лемченко К.П. Комп’ютерне моделювання: Навчальний посібник / Нікіфоров Т.Р. - МАУП, 2014. – 203 с.
9. Павленко П.П., розробка та тестування програмного забезпечення: Навчальний посібник / Павленко П.П. – КПІ, 2015. – 213 с.
10. Гончаренко М.Ф., інженеринг із захисту інформації: Навчальний посібник / Гончаренко М.Ф. – КПІ, 2012. – 190 с.