

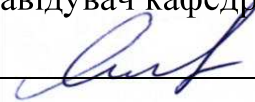
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра інформаційних управляючих систем та технологій

"Допущений до захисту"

Завідувач кафедри ІУСТ



к.т.н, доц. Михелєв І.Л.

" ____ " _____ 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти "магістр"

**на тему: «Системний аналіз методів та розробка СППР в медичній
діагностиці»**

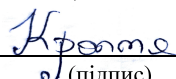
Виконав: студент групи 6145 м



Оселедько Н.А.
(підпис)

Керівник роботи:

доцент кафедри ІУСТ, к.т.н.,
(посада, науковий ступінь вчене звання)



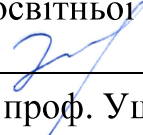
Кроптя А.В.
(підпис)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова
Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра інформаційних управляючих систем та технологій
Спеціальність 124 "Системний аналіз"
Освітня програма "Системний аналіз"

"ЗАТВЕРДЖУЮ"

Гарант освітньої програми


д.ф-м.н, проф. Ушкац М.В.

"_____" "_____" 2024 р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ на здобуття ступеня вищої освіти "магістр"

Студенту Оселедьку Нікіті Андрійовичу _____
(прізвище, ім'я, по батькові)

1. Тема роботи: Системний аналіз методів та розробка СППР в медичній діагностиці _____

Керівник роботи Кроптя А.В. _____

Затверджені наказом ректора № 1055-уч. від "08" жовтня 2024 року.

2. Термін подання роботи: 15 грудня 2024 р. _____

3. Вихідні дані до проекту (роботи) ДСТУ щодо обробки інформації, літературні джерела, технічна документація на існуючі аналоги систем, матеріали наукового стажування _____

4. Перелік питань, що належать до розробки (найменування розділів) _____

5. Перелік презентаційних матеріалів _____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Кроптя А.В.	05.09.2023	22.12.2023
2	Кроптя А.В.	23.12.2023	15.06.2024
3	Кроптя А.В.	15.06.2024	25.09.2024
4	Кроптя А.В.	25.09.2024	14.11.2024
5	Кроптя А.В.	25.09.2024	14.11.2024
6	Кроптя А.В.	25.09.2024	14.11.2024

7. Дата видачі завдання 05 вересня 2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів роботи	Примітка
1	Дослідження та аналіз предметної галузі	22.12.2024	
2	Розробка концепції	15.03.2024	
3	Розділ з розробки проектних рішень по системі (та її частинах)	15.06.2024	
4	Реалізація проекту та розробка робочої документації	25.09.2024	
5	Обчислення економічної ефективності	14.11.2024	
6	Вирішення питань охорони праці та навколишнього середовища	14.11.2024	
7	Оформлення пояснювальної записки	14.11.2024	
8	Подання магістерської роботи на перевірку на плагіат	16.11.2024	
9	Подання магістерської роботи рецензенту	20.11.2024	
10	Подання магістерської роботи на державний захист	15.12.2024	

Студент


(підпис)

Оселедько Н.А.
(ПІБ)

Керівник роботи


(підпис)

Кроптя А.В.
(ПІБ)

АНОТАЦІЯ

Тема магістерської роботи «Системний аналіз методів та розробка СППР в медичній діагностиці».

З розвитком медичних інформаційних технологій стали актуальними питання створення СППР, призначених у складі медичних інформаційних систем (МІС) надати лікарю дієву оперативну допомогу при діагностиці захворювань. Об'єктом дослідження є модель постановки діагнозів. Предметом дослідження є методи підтримки прийняття рішень в якості інструменту для постановки діагнозу.

Метою дослідження є підвищення продуктивності праці та якості діагностики за рахунок впровадження СППР для постановки діагнозу. Для досягнення мети в роботі вирішено задачі аналізу процесу та методів постановки діагнозу, розробки математичної моделі, аналізу методів підтримки прийняття рішень та дослідження клінічних систем підтримки прийняття рішень.

Наукова новизна роботи полягає в адаптації методів систем підтримки прийняття рішень для постановки діагнозу.

На етапі дослідження використовувались методи системного аналізу, дослідження операцій, прийняття рішень та інше. На етапі розробки застосовувалась методологія проектування UML.

Ключові слова: системний аналіз, системи підтримки прийняття рішень, діагностика, медицина, методи прийняття рішень.

SUMMARY

The theme of the master's thesis is "Systematic analysis of methods and development of DSS in medical diagnostics".

With the development of medical information technologies, the emergence of urgent issues of the creation of the DSS, called in the medical information systems (MIS), to provide the doctor with effective operational assistance in the diagnosis of diseases. The object of the study is a model for diagnosis. The subject of the study is decision support techniques as a tool for diagnosis.

The aim of the study is to increase productivity and quality of diagnosis through the introduction of an ADHD for diagnosis. To achieve the goal, the tasks of analyzing the process and methods of diagnosing, developing a mathematical model, analyzing decision support methods and researching clinical decision support systems have been solved.

The scientific novelty of the work is to adapt the methods of decision support systems for diagnosis.

At the research stage, methods of system analysis, operations research, decision making, and others were used. At the development stage, the UML design methodology was used.

Keywords: system analysis, decision support systems, diagnostics, medicine, methods of decision making.

ЗМІСТ

Вступ

1 ДОСЛІДЖЕННЯ ТА АНАЛІЗ МЕТОДІВ ПОСТАНОВКИ ДІАГНОЗУ.

ПОСТАНОВКА ЗАДАЧІ

1.1 Обумовлення актуальності. Постановка проблеми

1.2 Дослідження процесу постановки діагнозу лікарем

1.2.1 Поняття діагнозу

1.2.2 Структура та методологія діагнозу

1.2.3 Міжнародна класифікація хвороб і її застосування при оформленні діагнозу

1.2.4 Інформаційна підтримка постановки діагнозу

1.3 Прийняття управлінських рішень в процесі постановки діагнозу

1.3.1 Вхідні дані для прийняття рішень

1.3.2 Математичний апарат в процесі постановки діагнозу

1.4 Дослідження системи підтримки прийняття рішень та їх застосування в вирішенні задачі постановки діагнозу лікарем

1.4.1 Поняття СППР

1.4.2 Класифікація СППР

1.4.3 Архітектура СППР

1.4.4 Методи та моделі представлення даних та знань

1.4.5 Вибір методів підтримки прийняття рішень для постановки діагнозу лікарем

1.5 Дослідження сучасних засобів автоматизації процесу постановки діагнозу

1.5.1 Види медичних рішень та підходи до їх прийняття

1.5.2 Основні підходи, інструменти та принципи розробки інтелектуальної системи підтримки прийняття рішень

1.5.3 Аналіз існуючих інтелектуальних систем прийняття рішень

2 РОЗРОБКА ПРОЕКТУ ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1 Розробка концепції СППР

2.2 Загальносистемні рішення

2.2.1 Розробка діаграми варіантів використання

2.2.2 Розробка діаграм переходів станів

2.3 Рішення з інформаційного забезпечення

2.4 Рішення з математичного забезпечення

2.5 Рішення з програмного забезпечення

2.5.1 Розробка схеми взаємодії програмних модулів

2.5.2 Розробка діаграми розгортання системи

3 РЕАЛІЗАЦІЯ ПРОЕКТУ ІНФОРМАЦІЙНОЇ СИСТЕМИ

4 РОЗРАХУНОК ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ ВПРОВАДЖЕННЯ СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ

4.1 Вступ

4.2 Розрахунок витрат на створення та експлуатацію інформаційної системи

4.3 Економічна ефективність розробки і впровадження системи

5 ОХОРОНА ПРАЦІ

5.1 Вступ

5.2 Аналіз шкідливих та небезпечних факторів

5.3 Розрахунок системи штучного освітлення приміщення відділу інформаційних технологій

5.4 Розробка заходів щодо зменшення впливу шкідливих та небезпечних факторів при роботі з персональним комп'ютером

5.4.1 Заходи щодо зменшення впливу

5.4.2 Техніка безпеки при роботі на комп'ютері

6 ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

6.1 Вступ

6.2 Забруднення навколишнього середовища

6.3 Розробка заходів щодо зменшення забруднення навколишнього середовища

Висновки

Список використаної літератури

ДОДАТОК А – Технічне завдання

ДОДАТОК Б – Текст програми

ДОДАТОК В – Технічне описання

ДОДАТОК Г – Програма та методика випробувань

ВСТУП

У даний час прийняття рішень, засноване на теоретичних і практичних знаннях, є основою будь-якого виду людської діяльності – від вирішення невеликих приватних завдань до проблем державного рівня. Сучасний рівень розвитку інформаційних технологій дозволяє ставити і вирішувати завдання створення та впровадження різних систем підтримки прийняття рішень (СППР), в тому числі, і для ідентифікації об'єктів або їх станів. Необхідність використання СППР в управлінні очевидна, оскільки швидкий розвиток і ускладнення економічних, соціальних, технічних зв'язків, наявність складних взаємозв'язків між окремими процесами і явищами призводять до різкого зростання труднощів прийняття рішень. Наслідки помилок стають більш значущими. Використання СППР дозволить зняти гостроту проблеми, мінімізувавши кількість допускаються помилок. Особливої актуальності прийняття вірних рішень має в медицині та охороні здоров'я, так як помилкові дії в тактиці чи стратегії лікування хворого можуть призвести до подальшої втрати здоров'я людини. З розвитком медичних інформаційних технологій стали актуальними питання створення СППР, призначених у складі медичних інформаційних систем (МІС) надати лікарю дієву оперативну допомогу при лікуванні хворого.

Метою дослідження є підвищення продуктивності праці та якості діагностики за рахунок впровадження СППР для постановки діагнозу.

Для досягнення вказаної мети були поставлені наступні задачі дослідження:

- 1) Аналіз процесу постановки діагнозу;
- 2) Проведення аналізу методів постановки діагнозу;
- 3) Розробка математичної моделі;
- 4) Проведення аналізу методів підтримки прийняття рішень;
- 5) Дослідження клінічних систем підтримки прийняття рішень.

Об'єктом дослідження є модель постановки діагнозів.

Предметом дослідження є методи підтримки прийняття рішень в якості інструменту для постановки діагнозу.

Наукова новизна роботи полягає в адаптації методів систем підтримки прийняття рішень для постановки діагнозу лікарем.

В першому розділі роботи проводиться дослідження процесу діагностики, методів постановки діагнозу, аналіз існуючих клінічних систем підтримки прийняття рішень та дослідження методів підтримки прийняття рішень.

Другий розділ присвячений розробці проекту інформаційної системи.

В третьому розділі здійснюється реалізація проекту інформаційної системи.

Четвертий розділ присвячений розрахунку економічної ефективності системи підтримки прийняття рішень.

В п'ятому проводиться аналіз шкідливих та небезпечних факторів, розрахунок системи штучного освітлення та розробка заходів щодо зменшення впливу шкідливих та небезпечних факторів.

В шостому розділі визначається зменшення забруднення навколишнього середовища.

1 ДОСЛІДЖЕННЯ ТА АНАЛІЗ МЕТОДІВ ПОСТАНОВКИ ДІАГНОЗУ. ПОСТАНОВКА ЗАДАЧІ

1.1 Обумовлення актуальності. Постановка проблеми

Сучасне людство накопичило колосальний обсяг медичних знань, який, на жаль, не використовується в повному обсязі в клінічній практиці лікаря. За даними досліджень бостонської клініки Brigham and Women's Hospital, сьогодні у світі налічується понад 10 тисяч захворювань, понад 3 тисячі лікарських препаратів, 300 різних радіологічних процедур і більше тисячі лабораторних досліджень [1]. Обробка цих даних виходить за рамки можливостей людського мозку, тримати в голові такий обсяг інформації абсолютно неможливо. Крім того, рішення часто мають прийматися в умовах цейтноту, а також беручи до уваги високу вартість кожної помилки (оскільки мова йде про лікування пацієнтів), проблема забезпечення лікарів сучасними засобами підтримки прийняття рішень стає все більш актуальною. Тому в багатьох європейських країнах основні акценти в даний час робляться на інформаційну підтримку лікаря, на створення нового інформаційного середовища його діяльності, бо саме це дозволяє підвищити якість лікування і його доступність.

У даний час прийняття рішень, засноване на теоретичних і практичних знаннях, є основою будь-якого виду людської діяльності – від вирішення невеликих приватних завдань до проблем державного рівня.

Сучасний рівень розвитку інформаційних технологій дозволяє ставити і вирішувати завдання створення та впровадження різних систем підтримки прийняття рішень (СППР), в тому числі, і для ідентифікації об'єктів або їх станів. Необхідність використання СППР в управлінні очевидна, оскільки швидкий розвиток і ускладнення економічних, соціальних, технічних зв'язків, наявність складних взаємозв'язків між окремими процесами і явищами призводять до різкого зростання труднощів прийняття рішень. Наслідки помилок стають більш значущими. Використання СППР дозволить зняти гостроту проблеми, мінімізувавши кількість допускаються помилок. Особливої актуальності

прийняття вірних рішень має в медицині та охороні здоров'я, так як помилкові дії в тактиці чи стратегії лікування хворого можуть призвести до подальшої втрати здоров'я людини. З розвитком медичних інформаційних технологій стали актуальними питання створення СППР, призначених у складі медичних інформаційних систем (МІС) надати лікарю дієву оперативну допомогу при лікуванні хворого.

1.2 Дослідження процесу постановки діагнозу лікарем

1.2.1 Поняття діагнозу

Успіх діяльності лікаря будь-якої спеціальності неможливий без високого професіоналізму та здібності до логічного і аналітичного мислення. Перераховані якості лікаря - необхідна основа його роботи в клінічній (поліклінічній) медицині та в патології. Знання основних положень теорії діагнозу, їх практичне застосування в повсякденній діагностичній діяльності лікаря будь-якої спеціальності особливо важливі в умовах страхової медицини, при якій відомчий контроль якості діагностики та лікування в лікувально-профілактичних установах, як відомо, доповнюється незалежною експертизою страхових медичних компаній.

Труднощі діагностичного процесу посилюються у зв'язку з мінливістю хвороб, зростанням частоти патології, пов'язаних з лікуванням та множинністю захворювань. У цих умовах формулювання діагнозу є часто складним завданням навіть при розкритті у конкретного хворого (померлого) причин і механізмів розвитку захворювання, його наочно-змістовної характеристики. Ускладнює ситуацію і відсутність єдиних уявлень навіть у представників однієї спеціальності по ряду вузлових питань теорії діагнозу, у тому числі і за принципами формулювання діагнозу.

Діагноз в клінічній медицині - це короткий висновок про суть захворювання і стан хворого. Діагноз має бути правильним, розгорнутим та раннім. В основу діагнозу покладено нозологічний принцип, що вимагає дати назву хворобі, виявленій у обстежуваного відповідно до існуючої номенклатури.

Питання про своєчасність встановлення клінічного діагнозу не просте. При плановій госпіталізації і проведенні первинного комплексу обстеження клінічний діагноз встановлюють, як правило, через 2-3 доби після надходження хворого; при терміновій госпіталізації - негайно. У обох ситуаціях він може бути попереднім і може бути змінений після комплексу лікувально-діагностичних заходів.

За способом побудови та обґрунтуванням виділяють два види діагнозу - прямий і диференціальний. Суть *прямого діагнозу* полягає в тому, що лікар, зібравши всі типові, характерні ознаки, розглядає їх з погляду лише одного передбачуваного захворювання. Суть *диференційного діагнозу* полягає в тому, що з ряду загальних ознак, характерних для різних захворювань, після встановлення відмінностей виключають те або інше захворювання. Диференціальна діагностика полягає в порівнянні даної конкретної клінічної картини з рядом інших з метою ідентифікації однієї з них і виключення останніх.

Ознаками в діагностиці хвороб можуть бути «симптом», «синдром», «симптомокомплекс», «клінічна картина». Ці ознаки розрізняються по своїй специфічності і ступеню спільності.

Симптом - це одинична (специфічний або неспецифічний) ознака, виявлена будь яким методом обстеження хворого.

Симптомокомплекс - неспецифічна комбінація симптомів.

Синдром - специфічна комбінація декількох внутрішньо взаємопов'язаних симптомів. Специфічний симптом, симптомокомплекс, синдром відносяться до особливих ознак.

Клінічна картина - сукупність симптомів та симптомокомплексів - є загальною ознакою хвороби.

Лише в окремих випадках при виявленні характерного або високоспецифічного симптому (симптомокомплексу) можна встановити достовірний діагноз. Значно частіше лікар має справу з сукупністю загальних неспецифічних симптомів і повинен витратити значні зусилля на їх аналіз. При цьому в діагностиці симптоми не повинні підсумовуватися механічно, а взаємопов'язуватися з урахуванням значущості кожного з них.

У схемі діагностичного процесу прийнято виділяти три фази:

1. Виявлення всіх симптомів захворювання за допомогою клінічного та лабораторного обстеження, включаючи і негативні симптоми. Це *фаза збору відомостей про захворювання* у конкретного хворого.

2. Аналіз виявлених симптомів, їх сортування, оцінка по ступені важливості і характерності; зіставлення з симптомами відомих захворювань. Це *фаза аналізу і диференціації*.

3. Формулювання діагнозу захворювання на основі виявлених ознак і об'єднання їх в логічне ціле - *фаза інтеграції і синтезу*.

Клінічне мислення - це специфічна розумова свідомо і підсвідомо діяльність лікаря, що дозволяє найефективніше використовувати дані науки, логіки і досвіду для вирішення діагностичних і лікувальних завдань відносно конкретного хворого. Основні форми клінічного мислення реалізуються за допомогою аналізу і синтезу.

Успіхи професійної діагностичної діяльності лікаря зрештою визначаються логіко-методологічними можливостями його мислення. Потреба в знанні лікарями логіки сьогодні особливо зростає, бо стає очевидним, що значна частина діагностичних помилок - це не стільки результат недостатньої медичної кваліфікації, скільки майже невідворотний наслідок незнання і порушення самих елементарних законів логіки. Ці закони для будь-якого виду мислення, зокрема лікарського, мають нормативний характер, оскільки вони відображають об'єктивну визначеність, відмінності і обумовленість явищ матеріального світу.

Основні правила логічно стрункого лікарського мислення розкриваються в чотирьох законах логіки - законах вивідного знання. *Закон тотожності* характеризує визначеність мислення. Послідовність мислення визначається *законом суперечності* та *законом виключення третього*. Довідність мислення характеризується *законом достатньої підстави*.

Вимоги логічного **закону тотожності** полягають в тому, щоб поняття про предмет дослідження (наприклад, про симптом, нозологічну одиницю) було точно визначене і зберігало свою однозначність на всіх етапах розумового процесу. Закон тотожності виражається формулою: « $A \in A$ ». При цьому під « A » можна розуміти будь-який динамічний або відносно стійкий об'єкт (процес, ознаку процесу), аби в ході роздумів обраний зміст уявлення про об'єкт

залишався постійним. У діагностичній практиці дотримання закону тотожності вимагає перш за все конкретності і визначеності понять. Підміни поняття, тези, що відображають обговорюване явище в його істотних ознаках, служить частою причиною безплідних дискусій серед фахівців різних профілів.

Закон суперечності вимагає послідовності в міркуваннях, усунення суперечливих, понять і оцінок. Цей закон виражається формулою: « $A \in B$ » та « $A \notin B$ », причому ці думки не можуть бути одночасно істинними. Порушення закону суперечності виявляється в тому, що думка істинна затверджується одночасно і нарівні з думкою протилежною. Подібна ситуація виникає у випадках, коли діагностична гіпотеза ґрунтується на частини клінічної симптоматики, а інші ознаки захворювання, що суперечать висловленій думці, не враховуються.

Закон виключення третього, витікає із закону суперечності. Виражається формулою: « $A \in \text{або } B, \text{ або не } B$ ». Цей закон свідчить, що два суперечащих одне іншому вислови про один і той же предмет не можуть одночасно бути паралельно істинними чи паралельно помилковими. В цьому випадку з двох думок вибирають одну - істинну, оскільки третьої проміжної думки не існує. Наприклад, пневмонія в певних умовах може бути або основним захворюванням, що привело хворого до смерті, або тільки ускладненням інших захворювань.

Логічний закон достатньої підстави виражається формулою: «Якщо B , та в якості його підстави $\in A$ ». Закон свідчить, що всяка думка, щоб бути істинною, повинна мати достатню підставу. Обґрунтованість діагнозу спирається на встановлення специфічних для даної нозологічної форми симптомів та синдромів, які у свою чергу також мають бути обґрунтованими. Для обґрунтування діагнозу використовують перевірені практикою істини сучасної медичної науки. Порушення закону достатньої підстави продовжує залишатися джерелом суперечностей в деяких сучасних уявленнях про патогенез ряду захворювань, а також утруднень, пов'язаних з відтворюваністю одного і того ж клінічного і патологоанатомічного діагнозу різними фахівцями.

Приступаючи до складання діагнозу, кожний лікар повинен пам'ятати, що це не формальний акт, а укладений в чітку словесну формулу результат аналізу

клінічних симптомів, що спостерігалися у хворого, морфологічних змін в органах та тканинах в динаміці і взаємозв'язку.

Якнайповнішим і прийнятнішим можна вважати наступне визначення діагнозу в медицині:

Діагноз - медичний висновок про стан здоров'я обстежуваного (хворого), про наявне захворювання (травму) або про причину смерті, виражене в термінах, передбачених прийнятими класифікаціями і номенклатурою хвороб. Змістом діагнозу можуть бути також особливі фізіологічні стани організму, наприклад вагітність, клімакс, а також висновок про епідемічне вогнище».

1.2.2 Структура та методологія діагнозу

Прийнято виділяти 4 основних виду діагнозу в медицині: клінічний, патологоанатомічний, судово-медичний, епідеміологічний.

Повноцінними клінічний і патологоанатомічний діагнози можуть бути лише в тих випадках, коли при їх формулюванні враховується вся багатогранність їх функціонального навантаження.

У зв'язку з цим повноцінний клінічний діагноз повинен сприяти:

- комплексному лікуванню і вторинній профілактиці;
- своєчасному проведенню протиепідемічних заходів;
- медичній реабілітації;
- медичному прогнозуванню;
 - експертизі працездатності, професійному відбору і лікарському контролю в спорті;
 - експертизі придатності до військової служби, юридичної дієздатності;
- статистичному обліку захворюваності і смертності;
- навчанню клінічному мисленню і його вдосконаленню;
 - науковому аналізу питань клініки, діагностики, терапії та патоморфозу захворювань.

Повноцінний патологоанатомічний діагноз повинен сприяти:

- об'єктивному встановленню характеру, суті і походження патологічних процесів, станів і захворювань,

визначенню давності і послідовності їх виникнення та форми, ступеню розвитку і зв'язку між ними;

- визначенню причини і механізму смерті хворого;
- уточненню статистики смертності населення відповідно до Міжнародної класифікації хвороб;
- контролю за якістю клінічної діагностики та лікувального процесу;
- своєчасному проведенню протиепідемічних заходів;
- навчанню і вдосконаленню лікарського мислення;
- науковому аналізу питань етіології, патогенезу, патологічної анатомії, патоморфозу захворювань.

Клінічний і патологоанатомічний діагнози, будучи цілісною системою, представляють її на різних етапах і з дещо різних позицій. Якщо клінічний (прижиттєвий) діагноз є динамічним і по ходу спостереження за хворим може змінюватися, то остаточний (посмертний) клінічний і патологоанатомічний діагнози статичні, піддаються більшому логічному осмисленню, а отже, і вимоги до них інші.

Підхід до визначення основного захворювання в прижиттєвому і посмертному діагнозах різний. Наприклад, хворий може тривало лікуватися від цирозу печінки, а померти від інфаркту міокарду. У такому разі в посмертному клінічному діагнозі основним захворюванням буде інфаркт міокарду.

Відмінністю клінічного діагнозу від патологоанатомічного є те, що перший відбиває в основному «функціональні зміни», хоча може використовувати і морфологічні відомості (висновок біопсії), другий же базується в основному на морфологічних аспектах і використовує клінічні дані тільки у разі потреби (коли морфологічні зміни недостатньо чіткі або не встигли розвинутиися, або ліквідовані при лікуванні). При цьому в дужках указується, що ці дані внесені в патологоанатомічний діагноз з матеріалів історії хвороби.

Будучи клініко-морфологічними, володіючи незрівнянно більшою роздільною здатністю, патологоанатомічний діагноз має бути повним, тобто відбивати всі виявлені зміни.

Отже, за наявності деяких специфічних відмінностей, що не мають принципового значення, клінічний, остаточний клінічний і патологоанатомічний діагнози мають бути засновані на ідентичних принципах:

- діагноз має бути нозологічним;
- має відповідати міжнародній класифікації хвороб;
- містить додаткову характеристику: клініко-анатомічну форму (синдрому), тип перебігу, ступінь активності, стадію, функціональні порушення;
- має бути патогенетично обґрунтованим;
- містить чітку структуру з уніфікованими рубриками;
- фактична, логічна обґрунтованість і достовірність;
- своєчасність і динамізм.

1.2.3 Міжнародна класифікація хвороб і її застосування при оформленні діагнозу

Міжнародна класифікація хвороб (МКХ) використовується для перетворення мовного формулювання діагнозів хвороб і інших проблем, пов'язаних із здоров'ям, в літеро-цифрові коди, які забезпечують зручність зберігання, витягу і аналізу інформації.

МКХ є практичною, договірною, а не чисто теоретичною класифікацією. У ній допущені компроміси між класифікаціями, заснованими на етіології, анатомічній локалізації, обставинам виникнення хвороби і так далі. Проте її застосування забезпечує в практичній роботі уніфікацію оцінки стану здоров'я, хворобливих проявів, причин смерті і на цій основі створення стандартів діагностики і лікування.

У МКХ 10-го перегляду виділено 21 клас захворювань. При формулюванні діагнозу і кодуванні встановленої нозологічної одиниці використовують її найменування в рубриках перших дев'ятнадцяти класів; рубрики XX класу застосовують як додатковий код при встановленні зовнішніх причин травм, отруєнь та деяких інших наслідків дії зовнішніх факторів.

Важливими представляються рекомендації МКХ-10 щодо формулювання клінічного діагнозу в тих випадках, коли до кінця епізоду надання медичної допомоги точний нозологічний діагноз не встановлено. У таких ситуаціях на висновок лікаря про суть страждання у пацієнта слід вписати найменування виявлених симптомів, синдрому, відхилень від норми, а не формулювати діагноз як «можливий», «сумнівний», «передбачуваний», «діагноз із знаком питання».

Завершальний клінічний і остаточний патолого-анатомічний діагнози, відрізняючись деякими специфічними рисами, складаються по схожих правилах і мають схожу структуру з уніфікованими рубриками.

Залежно від виявленого варіанту генезу хвороби і смерті лікар, що лікує, і патологоанатом використовують наступні основні схеми оформлення діагнозу (таблиця 1.1).

Таблиця 1.1

Структура завершального клінічного і патологоанатомічного діагнозів (по Г.Г. Автанділову, 1994, із змінами)

Варіант генезу захворювання та смерті	Розділ діагнозу
А. Монокаузальний	I. Основне захворювання II. Ускладнення, зокрема безпосередня причина смерті III. Супутні захворювання
Б. Бікаузальний	I. Комбіноване основне захворювання: а) два конкуруючі основні захворювання б) основне і фонове захворювання в) два поєднані захворювання II. Ускладнення, зокрема безпосередня причина смерті III. Супутні захворювання
В. Мультікаузальний	I. Поліпатії: а) етіологічно і патогенетично пов'язані декілька хвороб і станів б) випадкове поєднання декількох хвороб і станів (асоціація хвороб) II. Ускладнення, зокрема безпосередня причина смерті III. Супутні захворювання

Основне захворювання - нозологічна одиниця, що має в даний момент найбільш виражені прояви, загрожуючі здоров'ю і життю хворого, з приводу якого проводиться лікування.

У патологоанатомічному розумінні основному захворюванню (первинній причині смерті) відповідає страждання, яке само по собі або через свої ускладнення спричинило смерть.

Ускладнення основного захворювання - це такий патологічний процес, який патогенетично і/або етіологічно пов'язаний з основним захворюванням і істотно обтяжує його перебіг, стаючи нерідко причиною смерті.

Доцільно виділяти головне ускладнення, яке часто виступає в ролі безпосередньої причини смерті, і другорядні ускладнення. У діагнозі ускладнення повинні розташовуватися в патогенетичній послідовності і таким чином відображати динаміку хвороби.

Безпосередня причина смерті - це структурно-функціональні або метаболічні порушення в організмі, що викликали зрив гомеостазу та привели до летального наслідку. Звичайно це головне ускладнення основного захворювання.

Супутнім називається захворювання, яке етіологічно і патогенетично не пов'язано з основним захворюванням і його ускладненнями, не справляє на їх перебіг та розвиток жодного впливу і не сприяє настанню смерті.

Комбіноване основне захворювання відображає поєднання двох провідних страждань у хворого. Ці хвороби викликають новий патологічний стан і, взаємодіючи між собою, призводять до смерті (бікаузальний генез захворювання і смерті).

Виділяють:

- *конкуруючі захворювання* - дві нозологічні одиниці, що одночасно є у хворого, кожна з яких окремо могла б привести до смерті; як правило, вони не піддаються розділенню через спільності клініко-морфологічних проявів та танатогенезу;

- *поєднані захворювання* - дві нозологічні одиниці, що випадково збіглися в часі та топіці, кожна з яких окремо в даних умовах не могла привести до смерті, але в сукупності через спільні ускладнення вони стали причиною смерті;

- *фонове захворювання* грає істотну роль у виникненні і розвитку основного захворювання, хоча етіологія у нього інша; взаємодія основної нозологічної одиниці з фоновим захворюванням прискорює і обтяжує танатогенез.

Слід зазначити, що зміст поняття «фонове захворювання» у ряді ситуацій до теперішнього часу не є однозначним.

З представлених схем структури діагнозів витікає, що при мультикаузальному генезі хвороби і смерті три нозологічні одиниці і більше оцінюються клініцистом і патологоанатомом як основне захворювання.

При монокаузальному генезі хвороби і смерті основне захворювання і первинна причина смерті виступають як однозначні поняття.

1.2.4 Інформаційна підтримка постановки діагнозу

Для автоматизації процесу постановки діагнозу лікарем необхідна повна наявність передумов для повної автоматизації процесу діагностики хвороби. Відсутність хоча б однієї з них або неповнота даних вимагає залучення в процес постановки діагнозу лікаря, який на основі свого досвіду може приймати правильні рішення.

Розгляд процесу постановки діагнозу лікарем з використанням нотації IDEF0 та обліком загальних вимог процесного підходу дасть змогу побачити інформацію, що впливає на хід виконання задачі (рисунок 1.1).

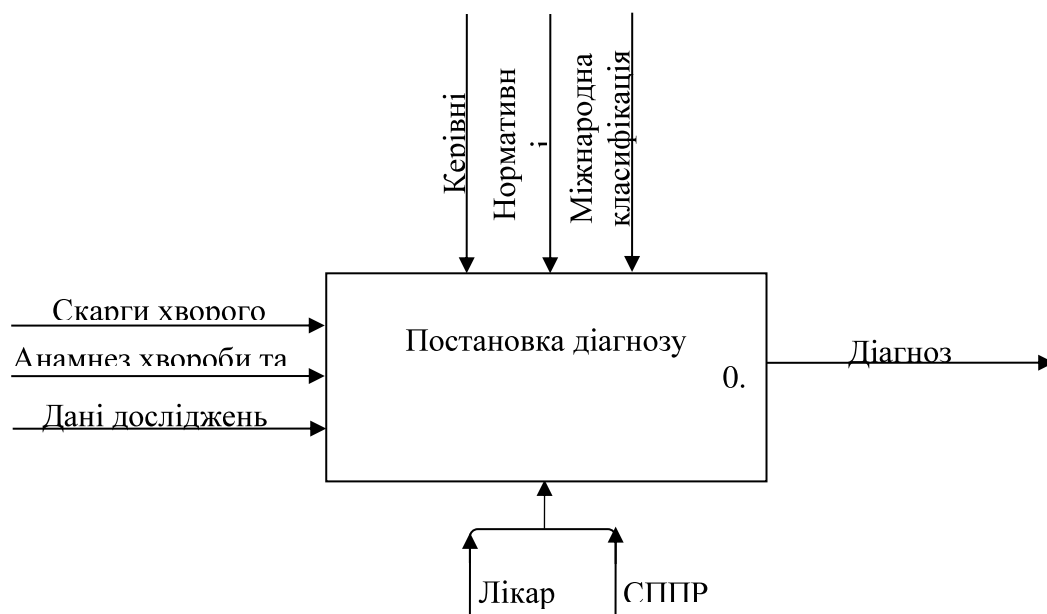


Рисунок 1.1 – Модель постановки діагнозу лікарем

Вхідні дані поділяються на наступні групи:

1) Початкові дані.

Диференційна діагностика повинна ґрунтуватися на:

а) ретельному вивченні скарг хворого,

б) анамнезу хвороби та життя,

в) даних:

- об'єктивного методу обстеження пацієнтів: загальний огляд пацієнта (загальний стан, положення в ліжку, свідомість, вираз обличчя, конституція, колір шкіри, стан волос та нігтів), пальпація (шкіри, підшкірної клітковини, лімфатичних вузлів), перкусія, аускультация;
- лабораторного методу дослідження: загальний аналіз, біохімічне та бактеріологічне дослідження. Особливості забору матеріалу в залежності від мети дослідження, підготовка пацієнта;
- інструментального методу дослідження: рентгенологічні, ендоскопічні, ультразвукові, радіоізотопні, комп'ютерні, у тому числі методи функціональної діагностики (ЕКГ, спірографія).

Саме такий підхід дозволить своєчасно поставити правильний діагноз і в абсолютної більшості хворих уникнути діагностичних помилок, а також виконання непотрібних операцій.

2) Керівні потоки служать обмеженнями при діагностиці. Лікар мусить враховувати наявність супутніх захворювань, мати в розпорядженні міжнародну класифікацію хвороб, нормативні (закони України, накази МОЗ України, ДСТУ та ін.) та керівні документи, а саме: довідники, на основі яких визначається діагноз;

3) Механізми, що необхідні для виконання процесу – лікар і СППР;

4) Результатом виконання роботи є готовий діагноз, який є основою для підбору лікування пацієнта.

Представлений склад даних інформаційного забезпечення постановки діагнозу дозволить створити єдину інформаційну середу, в результаті чого можна досягти підвищення рівня автоматизації діагностики захворювань.

1.3 Прийняття управлінських рішень в процесі проектування технологічних процесів

1.3.1 Початкові дані для прийняття рішень

Приступаючи до розробки працівник повинен мати вхідну інформацію.

Для постановки діагнозу вихідними даними є скарги хворого, повний анамнез хвороби та життя, результати об'єктивного (загальний стан, положення в ліжку, свідомість, вираз обличчя, конституція, колір шкіри, стан волос та нігтів, дані пальпації шкіри, підшкірної клітковини, лімфатичних вузлів, дані перкусія, дані аускультації), лабораторного (результати загального аналізу, біохімічного та бактеріологічного дослідження) та інструментального методу дослідження (висновки рентгенологічного, ендоскопічного, ультразвукового, радіоізотопного, комп'ютерного методів функціональної діагностики).

При діагностиці лікар повинен враховувати наявність супутніх захворювань. Крім того, лікар повинен мати у своєму розпорядженні наступне: нормативні, довідкові та керівні матеріали, міжнародну класифікацію хвороб і т.п.

1.3.2 Математичний апарат в проектуванні технологічних процесів

Основне завдання управління процесами проведення діагностики можна сформулювати наступним чином: дана сукупність інструментальних, клінічних та біохімічних засобів, в подальшому - засобів діагностичних технологій (ЗДТ) T_β , $\beta = 1, \dots, r$, які в результаті проведення досліджень $A = \{a\}$, забезпечують отримання первинної інформації про стан обстежуваного пацієнта. Протягом діагностики, первинна інформація, доповнюючись, внаслідок прийняття рішень (ПР), змінюється вторинної, в вигляді гіпотез про попередні, правдоподібні діагнози, потім - кінцевої, діагностичною інформацією, на підставі якої

приймається рішення про постановку клінічного або достовірного діагнозу. З огляду на особливості діагнозу, які полягають в необхідності використання як імовірнісних, так і детермінованих оцінок (попередній діагноз - імовірнісна оцінка, клінічний або заключний - достовірний), розглянемо в якості універсального розпізнає пристрої та системи ПР, інтелектуальний кінцевий автомат Finite-state machine - (FA). У рамках синтаксичного підходу, ідентифікацію стану пацієнта, як основний компонент технології постановки діагнозу, можна представити таким чином: граматика W , опис об'єкта в термінах «фрази» - X .

Така постановка задачі дозволяє визначити приналежність X до $L(W)$ для об'єктів з жорстко заданих правил функціонування. Кожній граматичній можна привести в відповідність якусь розпізнає пристрій у вигляді автомата. Розглянемо граматичну W , якій відповідає автомат з кінцевою безліччю станів. За визначенням, кінцевий автомат описується за допомогою п'яти характеристик:

$$AI = \{ Q, l, \delta, q_0, q_f \},$$

де Q - кінцева безліч станів; l - безліч додаються ознак; δ - таблиця переходів, що містить Ql можливих варіантів; q_0 - початковий стан ($q_0 \in Q$), q_f - безліч кінцевих станів, також що є підмножиною Q .

Залежно від можливих результатів розрізняють автомати детермінований і імовірнісний. Якщо перехід з одного стану в інший не залежить від імовірності можливих результатів і вхідного впливу l , стану q_0 , відповідає наступний стан q_f - можна говорити не про стохастичний, а детермінований кінцевий автомат. Кінцевий не детермінований автомат в діагнозі описується виразом

$$\delta(q_i, i_k) = \{ q_3, q_5, \dots, q_l \},$$

яке означає, що кожній парі величин (стан q_i і вхідний вплив i_k) відповідає один з наступних станів $q_3, q_5, \dots, q_l$. Наприклад, після отримання чергової ознаки, стан хворого можна узагальнити у вигляді попереднього діагнозу, який вибирається з безлічі правдоподібних гіпотез з різним ступенем ймовірності. Або, діагнозу хворого конкретного класу стану (стать, вік, фактори ризику та анти ризику), з певною ймовірністю відповідають причини розвитку захворювання. Детермінований кінцевий автомат застосовується в різних

завданнях діагностичного циклу, результатом яких може бути тільки одне рішення. У кожній з них, перехід має наступний вигляд:

$$g(q_i, i_k) = \{q_l\}$$

Наприклад, в задачах постановки діагнозу q_i - описується безліччю виявлених ознак, i_k - вхідний вплив є недостатньою ознакою x_{jj} , для постановки діагнозу D_j , q_l - виставляється діагноз D_j . За цією ж схемою, кожна обрана ознака, необхідна для постановки DS детермінує вибір методу визначення ЗДТ і т.д., згідно схеми оптимізації діагностики.

Для вирішення завдань діагностики, що характеризуються відсутністю повноти інформації використовуються методи математичного програмування, зокрема стохастичного програмування, що включають рішення в умови ризику та невизначеності.

Для вирішення завдань в умовах невизначеності ймовірності можливих результатів прийнятих рішень невідомі.

Наприклад, нехай є m допустимих рішень, n можливих результатів цих рішень і c_{ij} -витрати, пов'язані з рішенням i , $i = \bar{1}, m$, при виході j , $j = \bar{1}, n$. Істинний результат при прийнятті рішень i не відомий. Завдання полягає у виборі такого допустимого рішення i , яке в сенсі деякого критерію є найкращим. Якби обране рішення однозначно призводило до певного результату, то задача була б еквівалентна завданням лінійного або нелінійного програмування, так як необхідно вибрати таке рішення i при заданому результаті j , для якого величина c_{ij} , була б мінімальною. Але результат j не заданий, тому завдання вибору рішення ускладнюється. При рішенні задач в умовах ризику відомі ймовірності фіналів $p_1, \dots, p_n$, причому

$$\sum_{j=1}^n p_j = 1.$$

Для постановки достовірного діагнозу, необхідно розробити алгоритми побудови плану оптимальної діагностики, відповідно, до якої в СУП КД планувалися б цикли послідовного і цілеспрямованого визначення відсутніх ознак x_j $j = 1, \dots, N$ засобами діагностики для перекладу безлічі правдоподібних діагнозів в єдино достовірний.

1.4 Дослідження системи підтримки прийняття рішень та їх застосування в вирішенні задачі постановки діагнозу лікарем

В результаті необхідності спростити завдання пошуку рішення з'явився цілий клас програмних систем, покликаних полегшити роботу з аналізу даних. Такі системи прийнято називати системами підтримки прийняття рішень.

1.4.1 Поняття СППР

СППР - це комп'ютерна система, яка шляхом збору та аналізу великої кількості інформації може впливати на процес прийняття рішень організаційного плану в бізнесі та підприємстві.

СППР призначені, щоб допомагати проектувати, оцінювати альтернативи і контролювати процес реалізації. Основними функціями таких систем є:

- 1) надання допомоги ЛПР при аналізі вихідної інформації (оцінці сформованої обстановки і обмежень, що накладаються зовнішнім середовищем);
- 2) виявлення і ранжування пріоритетів, облік невизначеності в оцінках ЛПР і формування його переваг;
- 3) генерація можливих рішень (формування списку альтернатив);
- 4) оцінка можливих альтернатив, виходячи з переваг ЛПР, і обмеження, що накладається зовнішнім середовищем;
- 5) аналіз можливих наслідків прийнятих рішень;
- 6) вибір кращого, з погляду ЛПР, можливого варіанту.

Можна виділити три основні завдання, які вирішуються в СППР: введення, зберігання та аналіз даних. На СППР також можуть бути покладені завдання запобігання несанкціонованого доступу, резервного зберігання даних, архівування і т.п. Основним же завданням СППР є зіставлення кожної предметної ситуації з рекомендованими варіантами рішення.

Для завдань СППР властиві недостатність наявної інформації, її суперечливість і нечіткість, домінування якісних оцінок цілей і обмежень.

В якості інструментів узагальнення найчастіше використовуються засоби складання аналітичних звітів довільної форми, методи статистичного аналізу, експертних оцінок і систем, математичного та імітаційного моделювання. При цьому застосовуються бази узагальненої інформації, інформаційні сховища, бази знань про правила і моделі прийняття рішень.

1.4.2 Класифікація СППР

Класифікація СППР приведена на рисунку 1.2.



Рисунок 1.2 - Класифікація СППР

Системи підтримки прийняття рішень класифікують за:

- 1) взаємодією з користувачем:
 - а) пасивні допомагають у процесі прийняття рішень, але не можуть висунути конкретної пропозиції;
 - б) активні безпосередньо беруть участь у розробці правильного рішення;

- с) кооперативні припускають взаємодію СППР з користувачем. Висунуте системою пропозицію користувач може вдосконалити, а потім відправити назад в систему для перевірки. Після цього пропозиція знову представляється користувачеві, і так до тих пір, поки він не схвалить рішення;
- 2) способом підтримки:
- а) модельно-орієнтовані СППР, використовують в роботі доступ до статистичних, фінансових чи інших моделях;
 - б) СППР, засновані на комунікаціях, підтримують роботу двох і більше користувачів, які займаються спільним завданням;
 - с) СППР, орієнтовані на дані, що мають доступ до тимчасових рядів організації. Вони використовують в роботі не тільки внутрішні, а й зовнішні дані;
 - д) СППР, орієнтовані на документи, маніпулюють неструктурованою інформацією, укладеної в різних електронних форматах;
 - е) СППР, орієнтовані на знання, надають спеціалізовані рішення проблем, засновані на фактах;
- 3) сферою використання:
- а) загальносистемні працюють з великими системами зберігання даних і застосовуються багатьма користувачами;
 - б) настільні є невеликими системами і підходять для управління з персонального комп'ютера одного користувача
- 4) конкретним призначенням:
- а) оперативні, призначені для миттєвого реагування на поточну ситуацію - інформаційні системи керівництва (Executive Information Systems, ІСК);
 - б) стратегічні, засновані на аналізі великої кількості інформації з різних джерел з залученням відомостей, що містяться в системах, що акумулюють досвід вирішення проблем.

1.4.3 Архітектура СППР

Трьома головними компонентами СППР є підсистема інтерфейсу користувача, підсистема управління базою даних, підсистема управління базою моделей.

Компоненти СППР приведені на рисунку 1.3.

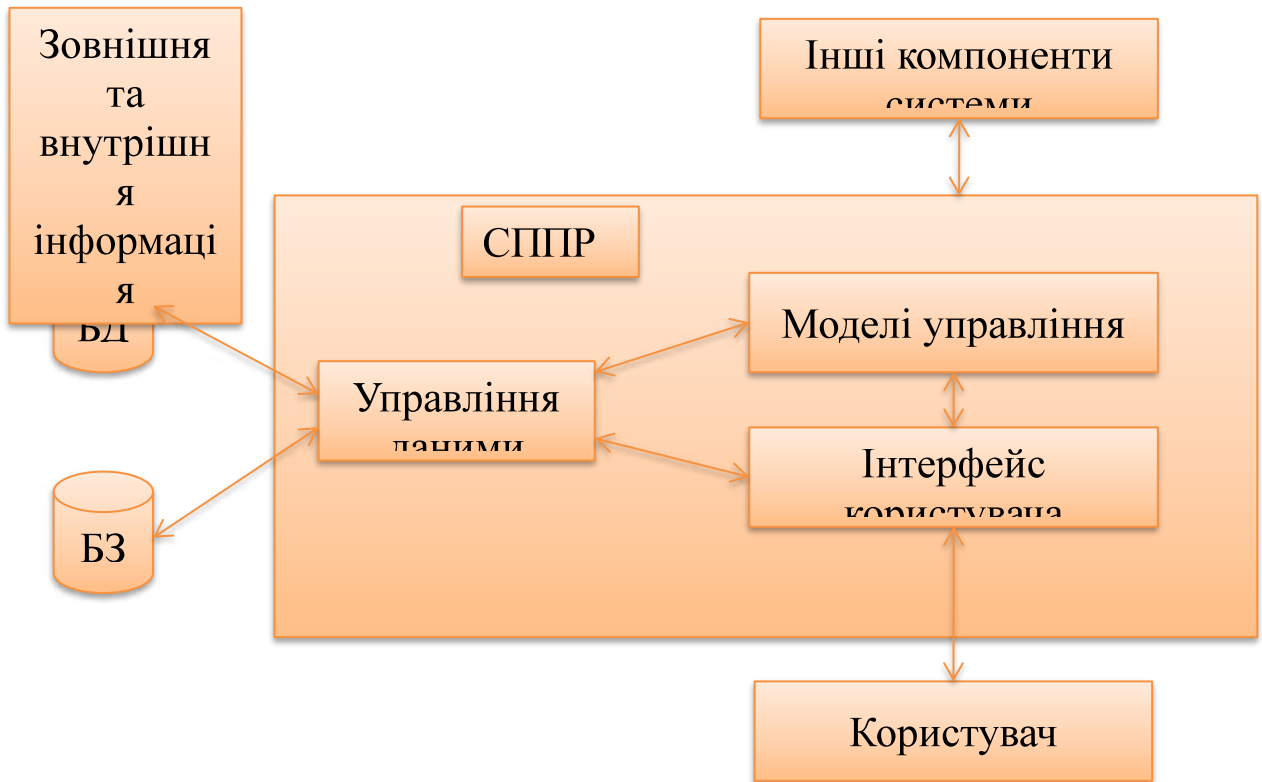


Рисунок 1.3 – Компоненти СППР

Архітектура СППР визначається характером взаємодії основних її складових — інтерфейсу користувача; бази та сховища даних, документів і правил; моделей і аналітичних інструментів; інфраструктури комунікацій і мереж, а також елементів цих частин. Ефективне поєднання всіх елементів СППР дає змогу уникнути ряду труднощів щодо побудови СППР і підвищити продуктивність комп'ютерної системи за рахунок:

- особливої інтеграції бази даних СППР з іншими внутрішніми і зовнішніми базами даних;
- скорочення тривалості очікування відповіді на запит користувача;
- ефективного використання великих математичних моделей;
- вдалішої координації діалогу з базою моделей та базою даних;
- поліпшення розуміння програмістами окремих аспектів системи;

- зниження витрат на створення та експлуатацію системи;
- максимального зменшення вартості підтримки та збільшення продуктивності користувачів, включаючи уникнення збоїв системи та інших проблем щодо продуктивності;
- зменшення інфраструктурних перешкод, які затримують розгортання нових додатків інформаційних систем і технологій, особливо СППР.

1.4.4 Методи та моделі представлення даних та знань

Знання - система понять і відносин для такого обміну. Можна умовно класифікувати знання в предметній області на понятійні, конструктивні, процедурні, фактографічні знання і мета знання.

Понятійні знання - набір понять, які використовуються при вирішенні даного завдання, наприклад, у фундаментальних науках і теоретичних областях наук, тобто це понятійний апарат науки.

Конструктивні знання - набори структур, підсистем системи і взаємодій між їх елементами, наприклад, у техніці.

Процедурні знання - методи, процедури (алгоритми) їх реалізації та ідентифікації, наприклад, у прикладних науках.

Фактографічні - кількісні та якісні характеристики об'єктів і явищ, наприклад, в експериментальних науках.

Мета знання про порядок і правила застосування знань (знання про знання).

Представлення знань є процес, кінцева мета якого - представлення інформації у вигляді інформативних повідомлень: фраз усного мовлення, пропозицій письмової мови, сторінок книги, понять довідника, об'єктів географічної карти, мазків і персонажів картини і т.п. Для цього необхідно користуватися певною конструктивною системою правил для їх подання і сприйняття.

Існують два типи методів представлення знань (ПЗ):

- 1) Формальні моделі ПЗ;
- 2) Неформальні (семантичні, реляційні) моделі ПЗ.

На відміну від формальних моделей, в основі яких лежить суворая математична теорія, неформальні моделі такої теорії не дотримуються. Кожна неформальна модель годиться тільки для конкретної предметної області і тому не володіє універсальністю, яка властива моделям формальним. У неформальних системах багато в чому визначається самим дослідником, який і відповідає за його коректність.

Кожному з методів відповідають свої моделі представлення знань (рисунок 1.4).

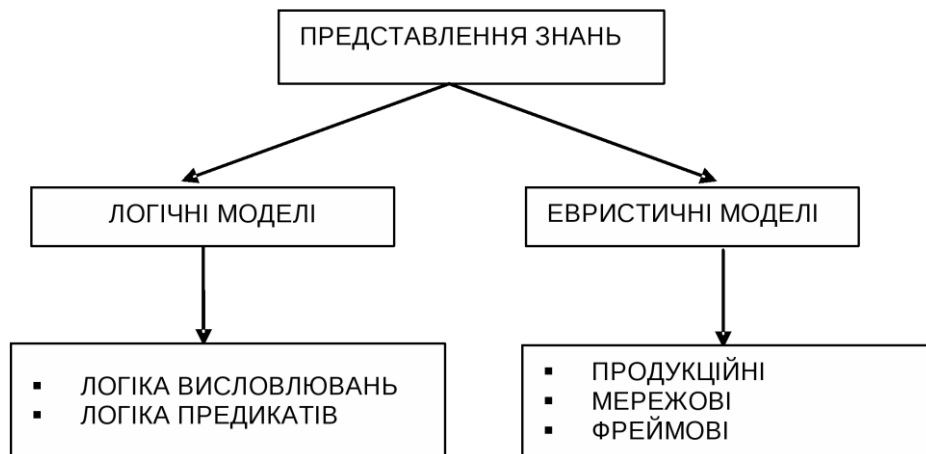


Рисунок 1.4 – Моделі представлення знань

Дамо загальну характеристику основних методів уявлень знань:

1) Логічні моделі представлення знань реалізуються засобами логіки предикатів.

Предикатом називається функція, що приймає тільки два значення – істина та неправда – і призначена для виразу властивостей об'єктів або зв'язків між ними. Вираз, в якому стверджується або не стверджується наявність яких-небудь властивостей у об'єкта, називається висловлюванням.

Константи служать для іменування об'єктів предметної області. Логічні речення або висловлювання утворюють атомарні формули. Інтерпретація предиката – це множина всіх допустимих зв'язувань змінних з константами. Зв'язування представляє собою підстановку констант замість змінних. Предикат вважається загальнозначущим, якщо він істинний на всіх можливих інтерпретаціях. Кажуть, що висловлювання логічно випливає із заданих посилок, якщо воно істинне завжди, коли істинні посилки.

Найбільш простою мовою логіки є обчислення висловлювань, в якому відсутні змінні. Будь-якому висловлюванню можна приписати значення *істина* або *неправда*. Окремі висловлювання можуть з'єднуватись зв'язками І, АБО, НІ, які називаються булевими операторами. Основи обчислень висловлювань складають правила утворення складних висловлювань з атомарних.

Головні переваги логічних моделей представлення знань полягає у можливості безпосередньо запрограмувати механізм виводу синтаксично правильних висловлювань. Однак з допомогою правил, що задають синтаксис мови, неможливо встановити істинність або неправду того чи іншого висловлювання. Причому це поширюється на всі мови. Висловлювання може бути побудовано синтаксично правильно, але бути зовсім беззмістовним.

2) Мережеві моделі. В основі моделей цього типу лежить конструкція, названа раніше семантичної мережею.

Залежно від типів зв'язків, які використовуються в моделі, розрізняють мережі, що класифікують, функціональні мережі і сценарії.

У мережах, що класифікують використовуються відносини структуризації. Такі мережі дозволяють у базах знань вводити різні ієрархічні відносини між інформаційними одиницями.

Функціональні мережі характеризуються наявністю функціональних відносин. Їх часто називають обчислювальними моделями, так як вони дозволяють описувати процедури «обчислень» одних інформаційних одиниць через інші.

У сценаріях використовуються каузальні відносини, а також відносини типів «засіб - результат», «знаряддя - дія» і т.п.

Якщо у мережевій моделі допускаються зв'язку різного типу, то її зазвичай називають семантичної мережею.

2) Продукційні моделі. У моделях цього типу використовуються деякі елементи логічних і мережевих моделей. З логічних моделей запозичена ідея правил виведення, які тут називаються продукціям, а з мережевих моделей - опис знань у вигляді семантичної мережі. У результаті застосування правил виведення до фрагментів мережевого опису відбувається

трансформація семантичної мережі за рахунок зміни її фрагментів, нарощування мережі та виключення з неї непотрібних фрагментів. Таким чином, у продукційних моделях процедурна інформація явно виділена і описується іншими засобами, ніж декларативна інформація. Замість логічного висновку, характерного для логічних моделей, у продукційних моделях з'являється висновок на знаннях.

Переваги продукцій:

- модульність;
- єдність структури (основні компоненти продукційної системи можуть використовуватись для побудови інтелектуальних систем з різноманітною проблемною орієнтацією);
- природність (вивід висновку в продукційній системі в багато чому аналогічний процесу роздумів експерта);
- гнучкість родовидової ієрархії понять, яка підтримується тільки як зв'язок між правилами (зміна правила не тягне за собою змін в ієрархії).

Недоліки продукцій:

- процес виводу менш ефективний, ніж в інших системах, оскільки більша частина часу при виводі витрачається на непродуктивну перевірку застосованості правил;
- цей процес важко піддається керуванню;
- складно представити родовидну ієрархію понять.

4) Фреймові моделі. На відміну від моделей інших типів у фреймових моделях фіксується жорстка структура інформаційних одиниць, яка називається протофреймом.

Значенням слота може бути практично що завгодно. Як значення слота може виступати набір слотів більш низького рівня, що дозволяє під фреймових уявленнях реалізувати «принцип матрьошки».

При конкретизації фрейму йому і слотам присвоюються конкретні імена і відбувається заповнення слотів. Таким чином, з протофреймов виходять фрейми

- екземпляри. Перехід від вихідного протофрейма до фрейму - примірнику може бути багатокроковим, за рахунок поступового уточнення значень слотів.

Найбільш яскраво переваги фреймових систем представлення знань проявляються в тому випадку, якщо родовидові зв'язки змінюються нечасто і предметна область нараховує небагато виключень. Значення слотів представляються в системі в єдиному екземплярі, оскільки включаються тільки в один фрейм, що описує найбільш загальні поняття з усіх тих, які містить слот з даним ім'ям. Така властивість систем фреймів забезпечує економне розміщення бази знань в пам'яті комп'ютера. Ще одна перевага фреймів полягає в тому, що значення будь-якого слота буде розраховано за допомогою відповідних процедур або знайдено евристичними методами. Тобто фрейми дозволяють маніпулювати як декларативними, так і процедурними знаннями.

До недоліків фреймових систем відносять їх відносно високу складність, що проявляється у зниженні швидкості роботи механізму виводу та збільшення трудомісткості внесення змін в родовидову ієрархію.

5) З інших методів представлення знань популярністю користується представлення знань по прикладам. Працюючи з системою такого типу, користувач задає їй декілька прикладів розв'язків задач з актуальної предметної області. На основі цих прикладів система самостійно будує базу знань, яка потім використовується для рішення інших задач. При створенні бази знань користувач має можливість у будь-який момент викликати на екран дисплея матрицю, що складається з прикладів задач та їх розв'язків, для того щоб встановити в ній наявність пустих місць, які необхідно заповнити прикладами «задача – рішення», яких не вистачає.

Знання в такій системі можуть зберігатися в різноманітних формах. Це може бути, наприклад, інтенціональна форма, коли користувач вводить в систему правила операції з атрибутами об'єктів предметної області, що приводять до необхідного рішення. Також це може бути екстенціональна форма, при якій кожен приклад детально описується користувачем і представляється в пам'яті комп'ютера у вигляді сукупності значень виділених атрибутів. Можлива комбінація і тої, і іншої форм. В результаті отримується матриця прикладів, яка

може бути розширена лише шляхом коректування прикладів, що містяться в матриці, або їх додаванням.

Основною перевагою представлення знань по прикладам є простота даного способу, оскільки користувач може не мати жодного представлення про продукційні правила, обчислення предикатів, фрейми та семантичні мережі. Разом з тим в якості недоліків метода представлення знань по прикладам відмічають відсутність гнучкості процесу побудови інтелектуальної системи. Користувач виявляється віддаленим від власноруч створеної бази знань і тому не може контролювати зв'язки між поняттями, що в ній містяться.

Вибір способу представлення знань виконується інженером по знанням після того, як ним досягнуто розуміння природи даних області, що моделюється. При розв'язку складних задач, коли джерела знань розрізняються по типам, представлення таких знань вимагає використання різних способів (змішане представлення). Тоді для продуктивного функціонування інтелектуальної системи нерідко застосовують принцип дошки об'яв, за допомогою якої реалізується взаємодія незалежних джерел знань.

Для побудови СППР та зберігання правил в базі знань було обрано продукційну модель представлення знань.

1.4.5 Вибір методів підтримки прийняття рішень для постановки діагнозу лікарем

При виборі комп'ютерних систем для розв'язання задач (готових чи розроблюваних на місці їх експлуатації) необхідно чітко зорієнтуватися в методологічній базі у відповідній предметній галузі перед тим, як починати аналіз спеціального програмного забезпечення чи готових систем СППР. Тому для проектування, розроблення та застосування СППР потрібна деяка узагальнена базова модель, яка була б зразком як для колективу розроблювачів, так і для кінцевих користувачів.

Рішення про те, який конкретно метод можна застосувати для підтримки прийняття рішень стосовно обраної задачі, є результатом отримання відповідей на ряд запитань. Центральним аспектом цього процесу вважається зіставлення

аналітичних задач та можливих методів їх розв'язання, які є в розпорядженні розроблювачів системи.

Для аналізу та вироблень пропозицій в СППР використовуються різні методи. Це можуть бути: інформаційний пошук, інтелектуальний аналіз даних, пошук знань в базах даних, міркування на основі прецедентів, імітаційне моделювання, еволюційні обчислення і генетичні алгоритми, нейронні мережі, ситуаційний аналіз, когнітивне моделювання та ін. Деякі з цих методів були розроблені в рамках штучного інтелекту.

Розглянемо детально кожен з них:

1) Інформаційний пошук (ІП) - процес пошуку неструктурованої документальної інформації та наука про це пошуку.

Пошук інформації являє собою процес виявлення в деякій множині документів (текстів) всіх таких, які присвячені зазначеної теми, задовольняють заздалегідь певній умові пошуку або містять необхідні факти, відомості, дані.

Процес пошуку включає послідовність операцій, спрямованих на збір, обробку та надання необхідної інформації зацікавленим особам.

У загальному випадку пошук інформації складається з чотирьох етапів:

- визначення (уточнення) інформаційної потреби і формулювання інформаційного запиту;
- визначення сукупності можливих власників інформаційних масивів (джерел);
- вилучення інформації з виявлених інформаційних масивів;
- ознайомлення з отриманою інформацією і оцінка результатів пошуку.

Методи інформаційного пошуку:

а) адресний пошук - процес пошуку документів з чисто формальними ознаками, зазначеним у запиті;

б) семантичний пошук - процес пошуку документів за їх змістом;

в) документальний пошук - процес пошуку в сховище інформаційно-пошукової системи первинних документів або в базі даних вторинних документів, що відповідають запиту користувача.

г) фактографічний пошук - процес пошуку фактів, відповідних інформаційним запитом.

2) Інтелектуальний аналіз даних (Data Mining) - виявлення прихованих закономірностей або взаємозв'язків між змінними у великих масивах необроблених даних. Підрозділяється на задачі класифікації, моделювання та прогнозування та інші.

Завдання, які вирішуються Data Mining:

- класифікація - віднесення вхідного вектора до одного з заздалегідь відомих класів;
- кластеризація - поділ множини вхідних векторів на групи (кластери) за ступенем «схожості» один на одного;
- скорочення опису - для візуалізації даних, лаконізму моделей, спрощення рахунку та інтерпретації, стиснення обсягів збираної і що зберігається;
- асоціація - пошук повторюваних зразків;
- прогнозування;
- аналіз відхилень;
- візуалізація.

3) Витяг знань у базах даних (KDD) - процес виявлення корисних знань в базах даних. Ці знання можуть бути представлені у вигляді закономірностей, правил, прогнозів, зв'язків між елементами даних та ін.

Головним інструментом пошуку знань у процесі KDD є аналітичні технології Data Mining, що реалізують завдання класифікації, кластеризації, регресії, прогнозування, передбачення і т.д.

KDD не ставить набір методів обробки або придатні для аналізу алгоритми, він визначає послідовність дій, яку необхідно виконати для того, щоб з вихідних даних отримати знання. Даний підхід універсальний і не залежить від предметної області, що є його безперечною гідністю.

4) Міркування на основі прецедентів.

Прецедент - випадок, що мав місце раніше і службовець прикладом або виправданям для наступних випадків подібного роду. Висновок на основі прецедентів (CBR) є підходом, що дозволяє вирішити нове завдання,

використовуючи або адаптуючи рішення вже відомої завдання. Як правило, такі методи міркувань включають в себе чотири основних етапи, які утворюють так званий цикл міркування на основі прецедентів чи CBR - цикл.

Основна мета використання апарату прецедентів в рамках СППР і, зокрема, систем експертної діагностики складних об'єктів, полягає у видачі готового рішення ЛПР для поточної ситуації на основі прецедентів, які вже мали місце в минулому при управлінні даними об'єктом або процесом.

5) Імітаційне моделювання - це метод дослідження, при якому досліджувана система замінюється моделлю з достатньою точністю описує реальну систему і з нею проводяться експерименти з метою отримання інформації про цю систему. Експериментування з моделлю називають імітацією.

Імітаційне моделювання - це окремий випадок математичного моделювання. Існує клас об'єктів, для яких з різних причин не розроблені аналітичні моделі, або не розроблені методи вирішення отриманої моделі. У цьому випадку математична модель замінюється імітатором або імітаційною моделлю.

Імітаційна модель - логіко-математичний опис об'єкта, яке може бути використане для експериментування на комп'ютері в цілях проектування, аналізу та оцінки функціонування об'єкта.

Мета імітаційного моделювання полягає у відтворенні поведінки досліджуваної системи на основі результатів аналізу найбільш істотних взаємозв'язків між її елементами.

Можна виділити два різновиди імітації:

- Метод Монте-Карло (метод статистичних випробувань);
- Метод імітаційного моделювання (статистичне моделювання).

Види імітаційного моделювання:

- Агентне моделювання - напрямок в імітаційному моделюванні, яке використовується для дослідження децентралізованих систем, динаміка функціонування яких визначається не глобальними правилами і законами (як в інших парадигмах моделювання), а навпаки.

- Дискретно-подієве моделювання - підхід до моделювання, що пропонує абстрагуватися від безперервної природи подій і розглядати тільки основні події модельованої системи, такі як: «очікування», «обробка замовлення», «рух з вантажем», «розвантаження» та інші. Цей вид моделювання найбільш підходить для моделювання виробничих процесів.

- Системна динаміка - парадигма моделювання, де для досліджуваної системи будуються графічні діаграми причинних зв'язків і глобальних впливів одних параметрів на інші в часі, а потім створена на основі цих діаграм модель імітується на комп'ютері.

6) Генетичні алгоритми - це евристичний алгоритм пошуку, використовуваний для вирішення завдань оптимізації та моделювання шляхом послідовного підбору, комбінування і варіації шуканих параметрів з використанням механізмів, що нагадують біологічну еволюцію. Є різновидом еволюційних обчислень.

Генетичні алгоритми служать, головним чином, для пошуку рішень в дуже великих, складних просторах пошуку.

Таким чином, можна виділити наступні етапи генетичного алгоритму:

1. Створення початкової популяції
2. Визначення (завдання) функцій пристосованості для особин популяції (оцінювання)

(Початок циклу)

1. Вибір індивідів з поточної популяції (селекція)
2. Схрещування і / або мутація
3. Обчислення функцій пристосованості для всіх особин
4. Формування нового покоління
5. Якщо виконуються умови зупину, то (кінець циклу), інакше (Початок циклу).

7) Штучні нейронні мережі (ШНМ) - математичні моделі, а також їх програмні або апаратні реалізації, побудовані за принципом організації та функціонування біологічних нейронних мереж - мереж нервових клітин живого організму. Це поняття виникло при вивченні процесів, що протікають в мозку при мисленні, і при спробі змоделювати ці процеси. Першою такою моделлю

мозку був перцептрон. Згодом ці моделі стали використовувати в практичних цілях, як правило, в задачах прогнозування.

Етапи рішення задач:

1. збір даних для навчання;
2. підготовка і нормалізація даних;
3. вибір топології мережі;
4. експериментальний підбір характеристик мережі;
5. експериментальний підбір параметрів навчання;
6. власне навчання;
7. перевірка адекватності навчання;
8. коректування параметрів, остаточне навчання;
9. вербалізація мережі з метою подальшого використання.

8)Методи штучного інтелекту.

Штучний інтелект (ШІ) - це наука про створення інтелектуальних машин і систем, особливо інтелектуальних комп'ютерних програм, спрямована на те, щоб зрозуміти людський інтелект. При цьому застосовуються методи не обов'язково повинні бути біологічно правдоподібні. Проблема полягає в тому, що невідомо які обчислювальні процедури ми хочемо називати інтелектуальними. А так як ми розуміємо тільки деякі механізми інтелекту, то під інтелектом в межах цієї науки ми розуміємо тільки обчислювальну частина здатності досягти мети в світі.

Якщо в основі роботи СППР лежать методи штучного інтелекту, то кажуть про інтелектуальну СППР, або ІСППР.

Галузі застосування методів підтримки прийняття рішень на основі інформаційних технологій приведені в таблиці 1.2.

Галузі застосування методів підтримки прийняття рішень на основі
інформаційних технологій

№	Метод ППР	Галузь застосування
1	Інформаційний пошук	Пошук документів, що задовольняють запиту, питання моделювання; Класифікація, фільтрація, кластеризація документів; Проектування архітектури пошукових систем і користувальницьких інтерфейсів; Вилучення інформації, зокрема анотування та реферування документів; мови запитів та ін.
2	Інтелектуальний аналіз даних	Нічим не обмежена
3	Витяг знань у базах даних	універсальний і не залежить від предметної області
4	Міркування на основі прецедентів	В рамках СППР і систем експертної діагностики складних об'єктів, полягає у видачі готового рішення ЛПР для поточної ситуації на основі прецедентів, які вже мали місце в минулому при управлінні даними об'єктом або процесом.
5	Імітаційне моделювання: 1) Агентне моделювання; 2) Дискретно-подієве моделювання; 3) Системна динаміка.	1) дослідження децентралізованих систем; 2) моделювання виробничих процесів; 3) побудова моделі бізнес-процесів, розвитку міста, моделі виробництва, динаміки популяції, екології та розвитку епідемії.
6	Генетичні алгоритми	Пошук рішень в дуже великих, складних просторах пошуку
7	Штучні нейронні мережі	Задачі прогнозування
8	Методи штучного інтелекту	Числова та символна обробка інформації Розпізнавання мови, образів, текстів

1.5 Дослідження сучасних засобів автоматизації процесу постановки діагнозу

1.5.1 Види медичних рішень та підходи до їх прийняття

Рішення в медицині - це фіксований медичний акт, який спрямовує діяльність медичного трудового колективу або окремого лікаря у русло досягнення цілей, поставлених перед медичною організацією або лікарем. До медичних рішень належать програмовані або не програмовані. Програмовані ті, де визначена наперед послідовність кроків у прийнятті рішень. Не програмовані ті, які виникають при нових ситуаціях або при невідомих факторах. Залежно від виду проблем та підходів їх вирішення у процесі формування медичних рішень, їх розділяють на інтуїтивні, основані на судженнях та раціональні рішення. Інтуїтивні медичні рішення - це вибір альтернативи чи варіанта на основі відчуття, що вона або він правильні. Рішення, засноване на судженнях - це вибір альтернативи на основі знань або досвіду. Раціональне медичне рішення - це вибір альтернативи за допомогою об'єктивного аналітичного процесу. До аналітичного процесу належать діагностика хвороб, формування критеріїв діагностики хвороб, скорочення числа альтернатив, оцінка альтернатив та вибір альтернатив. Основні види рішень в медицині представлені на рисунку 1.5.

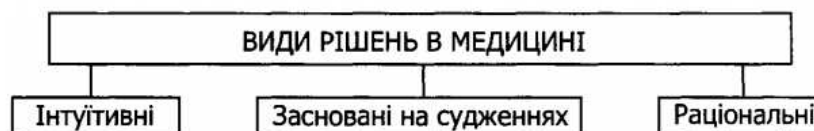


Рисунок 1.5 - Основні види рішень в медицині

У процесі прийняття медичних рішень лікарю необхідно відповісти на такі питання: що робити; як робити; кому доручити роботу; що це дає, тощо. Процес прийняття медичних рішень досить складний і багатогранний, а також залежить від кваліфікації лікаря, ситуації та ін. Найбільш визнаним підходом до прийняття медичних рішень є дотримання процедури виконання обов'язкових дій: розпізнавання проблеми; встановлення цілей розв'язання проблеми; вивчення проблеми за допомогою збирання та оброблення інформації, результатів аналізів, огляду хворого, медичної картки; обґрунтування альтернативних дій;

порівняння та відбір альтернатив; формулювання та видавання рішень (рисунки 1.6).

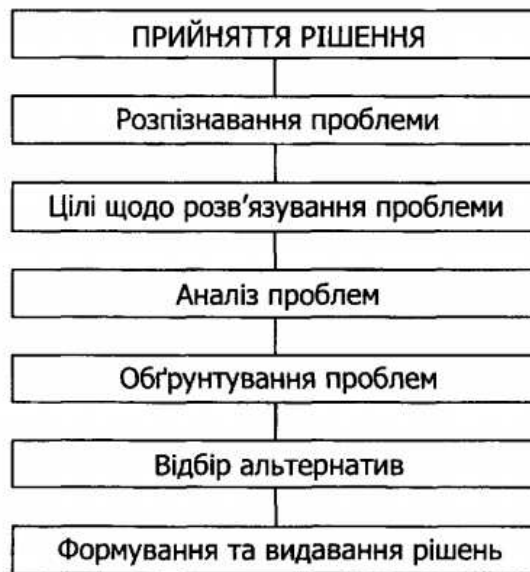


Рисунок 1.6 – Основні етапи процесу прийняття рішень

1.5.2 Основні підходи, інструменти та принципи розробки інтелектуальної системи підтримки прийняття рішень

Людський організм – це біологічна система з численними та ще не вивченими взаємозв'язками між елементами та середовищем проживання. Клініко-лабораторні показники хворого перебувають у складному взаємозв'язку один з одним. У більшості випадків про стан хворого можна судити не по значенням окремих показників, а по комбінаціям великої сукупності найбільш інформативних із них. Виявити такі показники і після поєднання їх значень «розпізнати образ» захворювання або ідентифікувати тяжкість стану хворого можна тільки з використанням методів системного аналізу, зокрема статистичного аналізу даних. Наразі повністю покладатися у прийнятті рішень на існуючі СППР не можна, адже більшість з них має певні недоліки, а саме: 1) «Інертність» і неможливість вносити зміни в систему. Даний недолік є найбільш значущим і перешкоджає повсюдного використання СППР в медицині, оскільки стає неможливим наповнювати систему новими знаннями про методики лікування та діагностики захворювань, лікарських препаратів та ін. 2) У необхідному обсязі не підтримується семантика медичних даних. Системи, засновані на жорстких правилах, не здатні забезпечити підтримку всього

комплексу взаємопов'язаних медичних знань, необхідних для прийняття рішення і формування плану лікування пацієнта. На практиці системи демонструють ряд істотних «обмежень». 3) Ізольованість від інших медичних систем, використовуваних в клінічній практиці. Системи підтримки прийняття рішень розробляються для медичних фахівців в окремо взятій предметній області. При цьому велика частина з них застосовується ізольовано від інших медичних систем і не використовує дані пацієнтів. 4) Посередня ергономіка і зручність системи для користувача. Інтерфейс більшості систем підтримки прийняття рішень є незручним для користувача, або вимагає значного часу на введення інформації про пацієнта. 5) Відсутність промислового впровадження систем. На сьогоднішній день як в Україні так і за кордоном немає відомих систем підтримки прийняття рішень, які б широко використовувалися в клінічній практиці, що викликано, в першу чергу, високою вартістю розробки та супроводу (в тому числі актуалізації) даних систем. 6) Відсутність можливості налаштування системи для використання в іншій предметній області. Системи підтримки прийняття рішень розробляються для медичних фахівців в окремо взятій предметній області і їх неможливо адаптувати до використання в іншій області без перепрограмування. При побудові медичних СППР необхідно враховувати наступні специфічні особливості: – людський організм надзвичайно складна функціональна система, про яку ми все ж знаємо мало. Через нестачу інформації, медичні знання мають досить складну структуру, що ускладнює їх формалізацію; – хвороби схильні до природної мінливості і, як правило, медичні задачі є слабо структурованими або неструктурованими проблемами; – не розроблені гнучкі і легко використовувані комп'ютерні методи машинного представлення медичних знань, а також відсутня формалізація процедури прийняття рішень; – системи діагностичного кодування стають в даний час більш універсальними, але детальна номенклатура ознак і симптомів, формати для реєстрації даних, а також організація записів визначаються індивідуально; – відсутня стандартизація в термінології, форматі, шкалах вимірювання медичних даних; – між хворими; страждаючими; одним захворюванням багато спільного, але кожен випадок є глибоко індивідуальний; – прийняття рішень в медицині безпосередньо пов'язане зі здоров'ям людини, тому рішення, отримані в СППР

повинні бути максимально достовірними. У медичних системах прийняття рішень, мають переважати ймовірнісно-статистичні методи, що дозволяють за допомогою довірчих інтервалів, рівнів значимості статистичних гіпотез та апостеріорних ймовірностей оцінювати достовірність отриманих рішень. При розробці та впровадженні медичних інформаційних систем має бути застосований системний підхід, який допоможе уникнути протиріч між складністю і придатністю системи [3]. При виконанні основних етапів системного аналізу вдасться уникнути надлишкової функціональності МІС, що в результаті призводить до непомірної складності розуміння системи для недосвідченого в інформаційних технологіях персоналу. Ефективність впровадження інформаційних технологій: стримується низкою проблем:

– необхідністю ручного введення первинної інформації в комп'ютер через клавіатуру. Рішення даної проблеми дозволить всі накопичені людством знання перевести в електронну форму, придатну для обробки та структуризації за допомогою комп'ютера; – наявність мовного бар'єру. Ця проблема вирішується різними способами, наприклад, створенням дружніх інтерфейсів, розробкою багато алфавітних інтерфейсів, автоматизованим перекладом; – невисоким рівнем розвитку мережових технологій; – низьким рівнем розвитку, інтелектуальних технологій; зокрема тих, що здатні підтримувати інтелектуальну діяльність людини. Інформатизація сфер суспільної діяльності дозволить вирішити проблему відчуження, накопичення, агрегування і збереження знань. Внаслідок людство отримає колективні знання, доступні будькому. «Колективні знання» – це феномен сучасної цивілізації.

На прикладі системи для ідентифікації ступеня тяжкості стану хворих побудова медичних СППР представлено як багатокроковий процес.

Перший етап – аналіз проблеми – включає осмислення медичної проблеми, пов'язаної з деяким захворюванням, її структуризацію, вивчення взаємозв'язку з іншими хворобами, оцінку повноти та достовірності інформації з даної проблеми, створення бази даних інформаційних ресурсів.

Другий етап – виявлення прихованих медичних знань за допомогою технологій аналізу даних. На даному етапі передбачається пошук і виявлення певних закономірностей в хаосі випадкових даних, та виділення в них сталих

зв'язків. Даний етап найбільш важливий для медицини, так як в процесі лікувальної діяльності накопичуються величезні масиви даних, в яких міститься — прихована інформація.

Третій етап — формулювання цілей і завдань — припускає визначення глобальної мети — лікування хворого від конкретного захворювання або сукупності захворювань, вироблення цілей і завдань, передбачуваних медичних заходів для досягнення глобальної мети.

Четвертий етап — визначення критеріїв — формує умови, у відповідності до яких можна зробити поділ хворих на групи однорідності по тяжкості стану.

П'ятий етап — формування безлічі альтернатив — груп однорідності хворих за ступенем тяжкості захворювання, діагнозу, видам захворювання і т. д.

Шостий етап — аналіз альтернатив — визначає належність хворого до тієї чи іншої групи однорідності. Даний етап є найбільш важливим, так як він є завершальним у процедурі підтримки прийняття лікарських рішень і полягає в оптимізації варіантів рішень (альтернатив) з вибором найкращого з безлічі згенерованих варіантів.

1.5.3 Аналіз існуючих інтелектуальних систем прийняття рішень

Використання в медичній практиці комп'ютерних технологій у поєднанні з вимірювальною та керуючою технікою дало змогу створити нові ефективні засоби для забезпечення автоматизованого збирання відомостей про стан хворого, їх опрацювання в реальному часі та моніторингу стану пацієнта. Цей процес привів до створення клінічних систем, які підняли на якісно новий рівень інструментальні методи досліджень та інтенсивну терапію.

Клінічні системи орієнтовані на виконання того чи того суворо визначеного кола типових медичних методик.

Комп'ютерні програми, як правило, пристосовані для розв'язання точно визначеного кола задач. Розширити чи якось змінити це коло можна тільки шляхом внесення у програму відповідних інструкцій програміста. Але таке вдосконалення потребує багато часу, до того ж завжди існує імовірність додаткових помилок. Розвиток технологій штучного інтелекту (програмних

засобів, які можуть у певних межах імітувати на комп'ютері деякі риси мислення людини) зумовив появу нового класу програмних засобів, здатних до самонавчання та накопичення нової, корисної інформації. Особливістю такої системи є поділ на базу даних, що містить історії хвороб, базу знань про предметну галузь і на механізм логічного висновку. Загалом інтелектуальна система припускає, що існує програма одержання знань, яку використовуватимуть для побудови і підтримки бази знань. Така система може також містити модуль пояснення, що допомагає пояснювати її рекомендації користувачам. Експертні системи широко застосовують у медицині для підтримки прийняття рішень при вирішенні різноманітних проблем діагностики, прогнозування, лікування, управління, навчання тощо.

Експертні системи в медицині мають задовольняти такі вимоги: моделювати поведінку кваліфікованого лікаря при розв'язанні діагностичної задачі, моделювати методи пошуку розв'язків; представляти здобуті розв'язки в такому вигляді, щоб вони були зрозумілі і лікарю, і хворому; швидко й порівняно просто адаптуватися до змін сукупності медичних знань, модифікуватися в разі появи нових або уточнення старих.

Існує багато видів експертних систем, серед яких можна виділити:

- експертні системи *інтерпретації* даних, що визначають зміст даних, зокрема даних медичних спостережень та дослідів;
- експертні системи *діагностики*, під час якої визначається характер відхилення стану об'єкта від норми і на цій основі його зараховують до відповідної категорії;
- експертні системи *прогнозування* роблять імовірнісні висновки про майбутній перебіг подій із ситуацій, що склалися, з урахуванням усіх обставин. У медицині з їх допомогою прогнозують перебіг хвороби при різних схемах лікування, визначаючи найкращу для конкретного хворого.

Повну історію хвороби, дані про аналізи за період лікування, стан організму на певний час – все це може зібрати і структурувати система штучного інтелекту. Дані, завантажені до бази, не залишаються поза увагою електронного мозку і будуть швидко оброблені. Це заощадить час лікарів, сприятиме

встановленню точного діагнозу і дозволить своєчасно призначити потрібне лікування.

Так працює система від IBM під назвою Watson Health. Вона здатна виявити потенційні проблеми з судинною системою, розпізнати рак, визначити, чи схильний пацієнт до утворення тромбів. IBM Watson може блискавично реагувати на те, коли необхідно вивчити нову інформацію і зробити висновки на її основі. Так, штучний інтелект від IBM за 10 хвилин проаналізував 20 млн наукових статей про онкологію і на їхній основі поставив пацієнту правильний діагноз.

У деяких лікарнях Великобританії вже використовується схожа розробка від Google – DeepMind Health. Ця розробка допомагає опрацювати всю інформацію про здоров'я людини, ділиться власними висновками зі спеціалістом, який лікує, і в результаті ставить остаточний діагноз.

Безпосередньо спілкуватися з людиною і щось рекомендувати можуть системи на зразок Ada – це сервіс, розроблений британською однойменною компанією. Медичний додаток спілкується з пацієнтом, розпитує про симптоми і скарги, а у відповідь надає рекомендації, у тому числі й стосовно лікаря, якого необхідно відвідати, а також пропонує зв'язатися з фахівцем, з метою віддаленої консультації.

Програма на основі III Sense.ly стежить за станом людей, які нещодавно пройшли тривале лікування або страждають від хронічних захворювань. Вона була розроблена стартапом з Сан-Франциско і зібрала 8 млн венчурних інвестицій на початковому етапі. Додаток створено для того, щоб структурувати дані про стан людини, він направляє їх фахівцям і надає рекомендації. Також система здатна нагадувати про час прийому ліків, необхідність відвідати лікаря.

Подібно функціонують і системи генетичного аналізу, який допомагає зрозуміти первинну причину захворювання. Одна з платформ щодо перевірки генома людини – Sophia Genetics – виявляє схильність пацієнта до різноманітних захворювань і звертає на це увагу лікаря.

2 РОЗРОБКА ПРОЕКТУ ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.2 Розробка концепції СППР

Концепція систем підтримки прийняття рішень полягає в наступному: користувач, а саме технолог, взаємодіє з СППР через інтерфейс користувача - обирає окрему модель або набір даних, що необхідно використовувати, а потім СППР надає результати через той самий інтерфейс.

Існуючі СППР частіше за все не зв'язані з базами даних та наповнюються вручну та вимагають спеціалізованої кваліфікації. СППР використовують для аналізу великі об'єми різноманітних даних, а значить проблема достатності й своєчасності надання даних є однією з найважливіших, тому що відсутність або некоректність інформації приводить до спотворення результатів аналізу. Все вищезазначене не дозволяє використовувати концепцію СППР в чистому вигляді для вирішення реальних промислових завдань.

Таким чином, вибір в якості стратегії розробки СППР при постановці діагнозу змушує реалізувати множину функцій, що не потрапляють під концепцію, але є необхідними для підтримки прийняття рішень: задачі збору, обробки й передачі інформації. Тому концепція прийме наступний вигляд (рисунок 2.1).

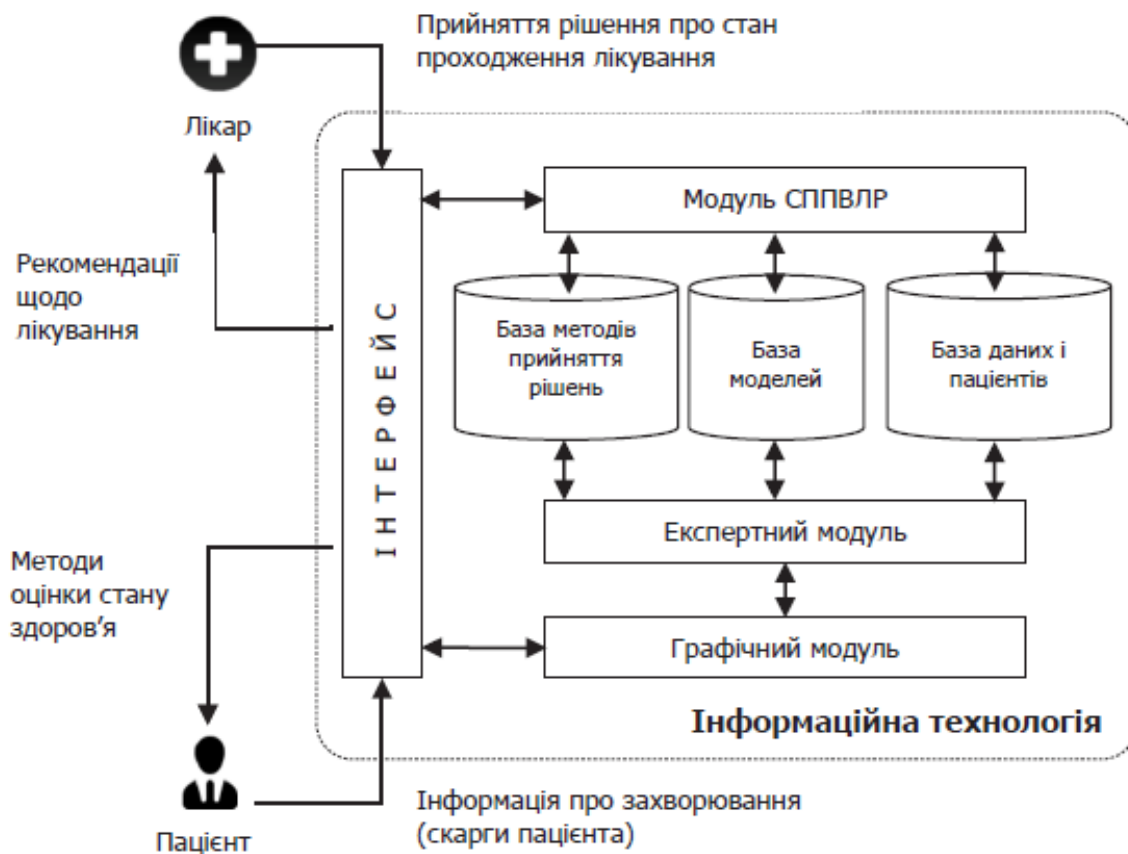


Рисунок 2.1 – Схема запропонованої концепції

2.2 Загальносистемні рішення

На етапі загальносистемних рішень було визначено загальний функціонал системи.

2.2.1 Розробка діаграми варіантів використання

Діаграма прецедентів є графом, що складається з множини акторів, прецедентів (варіантів використання) обмежених границею системи (прямокутник), асоціацій між акторами та прецедентами, відношень серед прецедентів, та відношень узагальнення між акторами. Діаграми прецедентів відображають елементи моделі варіантів використання.

Суть даної діаграми полягає в наступному: проектована система представляється у вигляді безлічі сутностей чи акторів, взаємодіючих із системою за допомогою так званих варіантів використання. Варіант використання (use case) служить для опису сервісів, що система надає актору. Іншими словами, кожен варіант використання визначає деякий набір дій,

чинений системою при діалозі з актором. При цьому нічого не говориться про те, яким чином буде реалізована взаємодія акторів із системою.

У мові UML є кілька стандартних видів відношень між акторами і варіантами використання: • асоціації (association relationship)

- включення (include relationship)
- розширення (extend relationship)
- узагальнення (generalization relationship)

При цьому загальні властивості варіантів використання можуть бути представлені трьома різними способами, а саме — за допомогою відношень включення, розширення і узагальнення.

Відношення асоціації – одне з фундаментальних понять у мові UML і в тій чи іншій мірі використовується при побудові всіх графічних моделей систем у формі канонічних діаграм.

Включення (include) у мові UML — це різновид відношення залежності між базовим варіантом використання і його спеціальним випадком. При цьому відношенням залежності (dependency) є таке відношення між двома елементами моделі, при якому зміна одного елемента (незалежного) приводить до зміни іншого елемента (залежного). Відношення розширення (extend) визначає взаємозв'язок базового варіанта використання з іншим варіантом використання, функціональна поведінка якого задається базовим не завжди, а тільки при виконанні додаткових умов.

Діаграма варіантів використання зображена на рисунку 2.2.

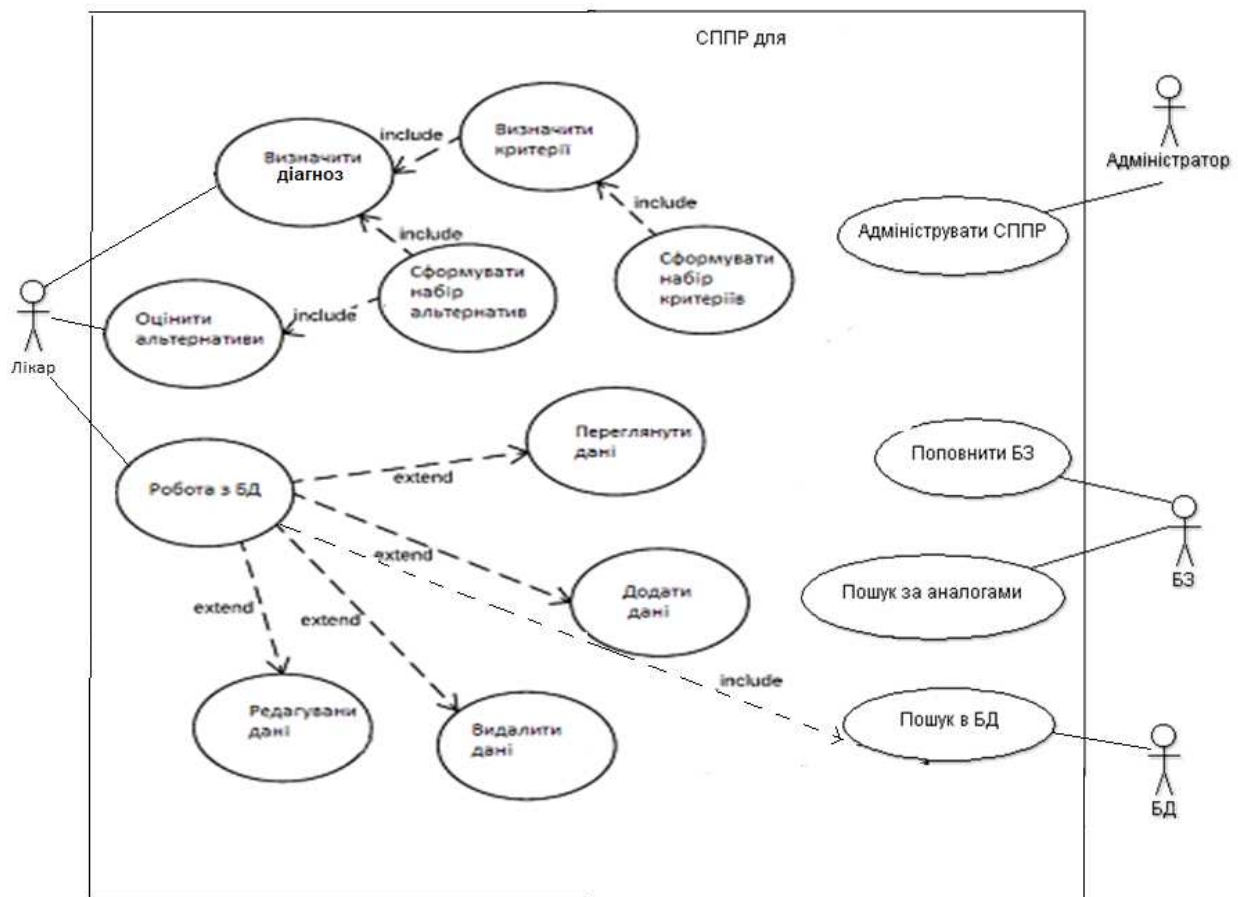


Рисунок 2.2 – Діаграма варіантів використання

Користування системою підтримки прийняття рішень для постановки діагнозу лікарем передбачає наявність зовнішніх бази даних та бази знань.

База даних містить в собі вхідну інформацію для прийняття рішень лікарем.

Безпосереднім користувачем СППР для проектування ТП є лікар. Для цього система йому надає наступні функції. Початковим етапом формування діагнозу за допомогою даної системи є «Пошук за аналогом», що означає за допомогою бази знань визначити чи є в базі аналог необхідного діагнозу. Цей процес розширює прецедент «Виробити альтернативні рішення», що є основним в процесі діагностики.

Коректну роботу системи забезпечує підтримка її адміністратором в належному стані – прецедент «Адмініструвати СППР». Для цього від має доступ до функцій, що недоступні іншим користувачам СППР.

2.3.2 Розробка діаграми переходів станів системи

Для опису поведінки системи була розроблена діаграма станів, яка показує реакцію системи на зовнішній вплив

Початком роботи користувача з системою є «Авторизація», яка потребує введення логіну та паролю. Без успішної авторизації користувач не може перейти до використання функцій системи. В разі успішної авторизації система переходить в стан «Очікування вибору дії користувача».

Процес постановки діагнозу являє собою діалог користувача з системою з поступовим уточненням потреб, вимог та обмежень до технологічного процесу. Після успішного «Введення даних» може бути два варіанти вибору подій «Обробка даних» або «Пошук аналогів».

Якщо обрана подія «Обробка даних», то наступним етапом є збереження даних в БД (скарги, дані обстежень і т.д.) – «Поповнення БД». Поповнення виконано, а значить повертаємось до стану «Очікування вибору користувача», де маємо можливість розпочати спочатку, вибрати іншу дію, або вийти з системи.

Якщо ж обрана подія «Пошук аналогів» та зроблений запит на пошук, то система управління зв'язується з базою знань та шукає вже створені технологічні процеси зі схожими ознаками. СУБЗ знайшла аналог, який потребує змін – перехід до стану «Корегування діагнозу», далі зберігає нові ТП – стан «Поповнення БЗ», після чого виконується «Вироблення альтернативних рішень». СУБЗ знайшла аналог, який не потребує змін, відповідно, виконується перехід до стану «Вироблення альтернативних рішень». СУБЗ не знайшла аналог, а значить треба створити новий діагноз – стан «Формування нового діагнозу», наступним етапом є збереження діагнозу - стан «Поповнення БЗ», після чого виконується «Вироблення альтернативних рішень». Вироблення виконано, а значить повертаємось до стану «Очікування вибору користувача», де маємо можливість розпочати спочатку, вибрати іншу дію, або вийти з системи.

Діаграма переходів станів системи представлена на рисунку 2.3.

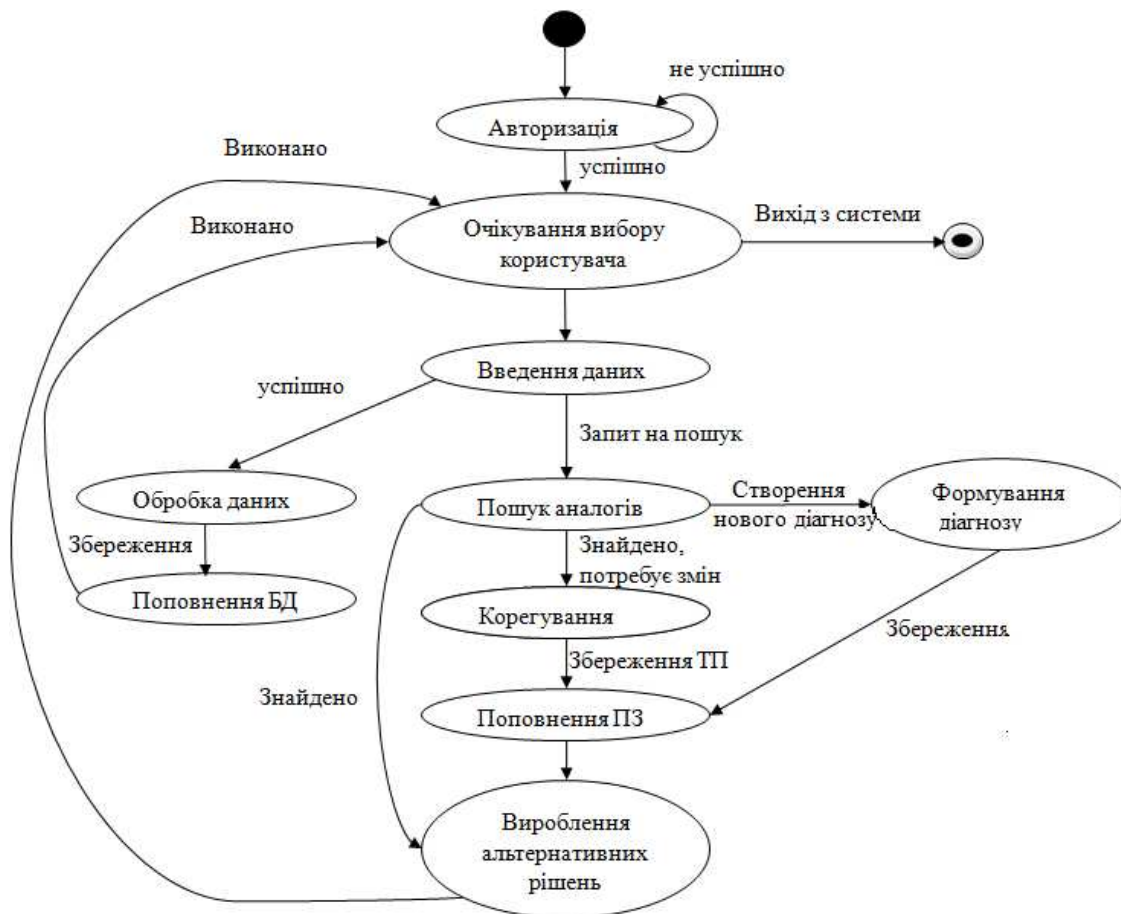


Рисунок 2.3 – Діаграма переходів станів системи

2.4 Рішення з технічного забезпечення

Технічне забезпечення - комплекс технічних засобів, засобів експлуатаційної підтримки і документація на ці засоби і технологічні процеси, внутрішні стандарти підприємства. Це:

- технічні засоби збору, реєстрації, накопичення, обробки, відображення, розмноження, доставки, збереження і забезпечення безпеки інформації;
- комп'ютери будь-яких моделей, потужні серверні і мережеві пристрої, оргтехніка;
- телекомунікаційна техніка і засоби зв'язку;
- загальносистемна документація, що включає державні, галузеві і корпоративні стандарти по технічному забезпеченню;

- спеціалізована документація, яка містить методичні матеріали по всіх етапах проектування, розробки, впровадження, супроводу та застосування технічних і технологічних засобів;
- нормативно-довідкова документація для виконання технічного забезпечення.

При проектуванні системи підтримки прийняття рішень був проведений аналіз структури і завдань, на які націлена дана система. На основі цього аналізу були зроблені висновки про склад і параметри технічних засобів, які будуть задіяні при використанні системи.

Технічне забезпечення СППР має включати в свій склад:

- серверний комплекс, інтегрований в інформаційно-обчислювальну мережу користувачів СППР;
- робочі станції, які повинні забезпечувати роботу користувачів на відповідних робочих місцях;
- пристрої для резервного копіювання та зберігання даних СППР;
- засоби виведення даних на друк;
- засоби безперебійного живлення.

Для реалізації системи підтримки прийняття рішень були обрані наступні компоненти:

- a) Windows 7 – в якості операційної системи;
- b) MySQL – в якості системи управління базами даних;
- c) Java – в якості засобів розробки.

2.5 Рішення з математичного забезпечення

Рішення з математичного забезпечення представлені математичною моделлю ідентифікації діагнозу в умовах невизначеності.

Задача ідентифікації діагнозу в умовах невизначеності має такий вигляд: вектором $s \in S_n$ позначимо вектор симптомів, який точно визначає діагноз $d_i \in D$, у векторі s довільним чином частково замінимо компоненти на $*$ – зазначимо їх, як невизначені однозначно, у результаті ми отримаємо деякий вектор s' . Можна вважати, що вектор s' містить інформацію про вектор s у неповному

обсязі. У такому разі, отримаємо задачу визначення набору компонент вектора симптомів s' , якого буде достатньо для однозначної ідентифікації діагнозу d_i .

Зазначимо через $M_s = \{m_1, m_2, \dots, m_{p-1}\}$ – допустиму послідовність ознак симптомів, тоді $q \in Q = \{m_1, m_2, \dots, m_{p-1}, q\}$ – довільна допустима послідовність.

Для будь-якого пацієнта, $P = \{P_i\}_{i=1}^p$, де $p = 1, k$, маємо:

$M_s \cap S(d_{i_1}, d_{i_2}, \dots, d_{i_p}) = \{i_p\}$, а $M_s \cap S(d_{i_1}, d_{i_2}, \dots, d_{i_{p-1}}, q) = \emptyset$, при $q = d_{i_p}$.

Тому, $\chi(s, S(d_{i_1}, d_{i_2}, \dots, d_{i_p})) = \frac{1}{|S(d_{i_1}, d_{i_2}, \dots, d_{i_p})|} > 0$, а $\chi(s, S(d_{i_1}, d_{i_2}, \dots, d_{i_{p-1}}, q)) = 0$, якщо $q = d_{i_p}$. Таким чином, $i_1, i_2, \dots, i_k$ – повна система вирішальних ознак для елемента d_i . Очевидно, що ця система ознак є мінімальною. Алгоритм ідентифікації точно встановить діагноз, при умові надання всього k . Число k на порядок менше числа ознак в еталонному векторі.

2.6 Рішення з програмного забезпечення

2.6.1 Розробка схема взаємодії програмних модулів

Схема взаємодії програмних модулів визначає загальну структуру програмного забезпечення системи та взаємодії її частин.

«Авторизація користувача» - надання певній особі або групі осіб прав на виконання певних дій; а також процес перевірки даних прав при спробі виконання цих дій.

«Головна форма» дозволяє вибрати одну з запропонованих дій.

Модуль «Робота з БД» забезпечує перегляд та поповнення бази даних.

Модуль «Управління процесом діагностики» передбачає формування, корегування діагнозу, виробку альтернативних рішень або пошук за аналогом.

Модуль «Поповнення БЗ» дозволяє зберігати створений або скорегований діагноз.

Схема взаємодії програмних модулів представлена на рисунку 2.4.

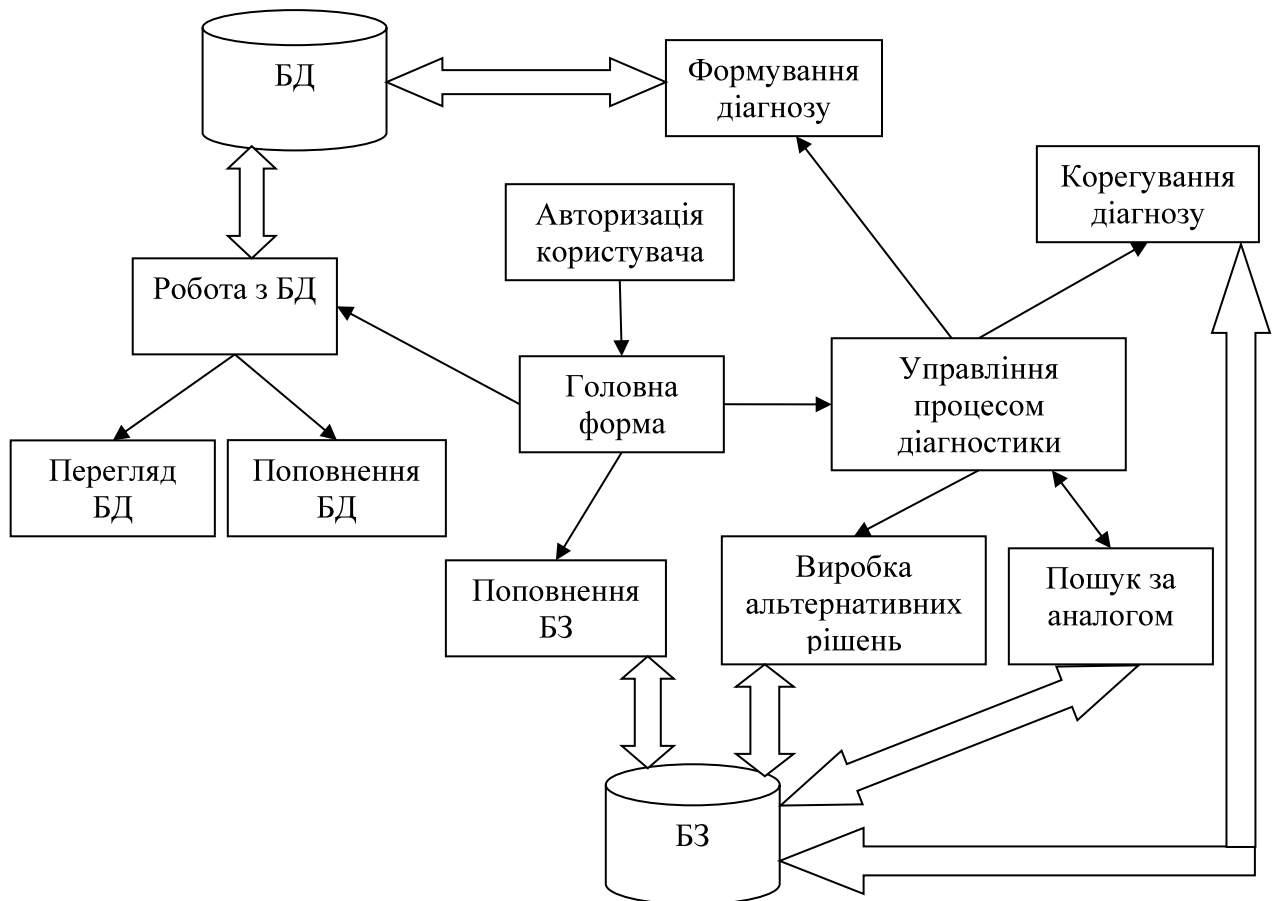


Рисунок 2.4 – Схема взаємодії програмних модулів

2.6.2 Розробка діаграми розгортання системи

Розробка діаграми розгортання являється останнім етапом специфікації моделі системи.

Діаграма розгортання показує конфігурацію вузлів, де проводиться обробка інформації, і те, які компоненти розміщені на кожному вузлі. Вузол, як і компонент, що існує в матеріальному світі і є важливим будівельним блоком при моделюванні фізичних аспектів системи. Вузол - це фізичний елемент, який існує під час виконання та представляє обчислювальний ресурс, зазвичай володіє як мінімум деяким об'ємом пам'яті, а часто також і процесором. Вузли використовуються для моделювання, топологія апаратних засобів, на яких виконується система. Як правило, вузол - це процесор або пристрій, на якому можуть бути розгорнуті компоненти.

Діаграма розгортання зображена на рисунку 2.5.

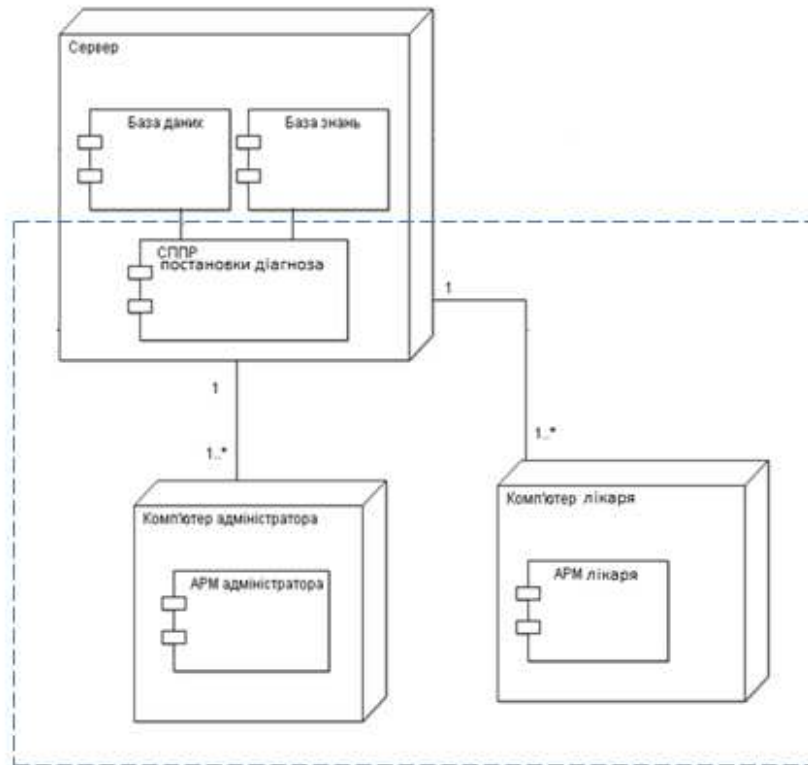


Рисунок 2.5 – Діаграма розгортання системи

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

На вже існуючому сервері підприємства де розгорнуто загальну базу та базу знань розгортається СППР для проектування ТП. Для її адміністрування залучається адміністратор – для цього на робочій станції в інформаційному відділі підприємства розгортається модуль адміністрування. Безпосереднім користувачем СППР являється технолог – на його робочій станції розгорнуто компонент «АРМ технолога» що надає функції по проектуванню ТП.

Загальна система також передбачає розгортку модуля конструктора, на його робочій станції, для поповнення бази даних електронними моделями деталей. Розгортку модуля для планового відділу для роботи з планом виробництва. А також розгортку модуля для відділу стандартизації для ведення нормативної інформації.

3 РЕАЛІЗАЦІЯ ПРОЕКТУ ІНФОРМАЦІЙНОЇ СИСТЕМИ

Система керування базами даних (СКБД) або Система управління базами даних (СУБД) — комплекс програмного забезпечення, що надає можливості створення, збереження, оновлення та пошуку інформації в базах даних з контролем доступу до даних. На сьогоднішній день існує багато різних СУБД: Firebird, Interbase, MS SQL Server, MySQL, Oracle Database і ін.

В якості СУБД була обрана MySQL, оскільки:

- MySQL - компактний сервер баз даних, який характеризується великою швидкістю, стійкістю і простотою використання;

- MySQL вважається гарним рішенням для малих і середніх додатків.

Можливості сервера MySQL:

- простота в установці і використанні;
- підтримується необмежена кількість користувачів, що одночасно працюють з БД;

- кількість рядків у таблицях може досягати 50 млн;

- висока швидкість виконання команд;

- наявність простої і ефективної системи безпеки.

Можливості MySQL Workbench:

- дозволяє наочно уявити модель БД в графічному вигляді;

- наочний і функціональний механізм установки зв'язків між таблицями, в тому числі «багато до багатьох» зі створенням таблиці зв'язків;

- Reverse Engineering - відновлення структури таблиць з уже існуючою на сервері БД;

- зручний редактор SQL запитів, дозволяє відразу ж відправляти їх сервера і отримати відповідь у вигляді таблиці;

- можливість редагування даних в таблиці у візуальному режимі.

Для написання програми було обрано мову програмування Java.

Java дозволяє створювати незалежні від платформи програми шляхом компіляції в проміжне представлення, яке називається байткодом. Багато попередніх спроб знати розв'язок проблеми незалежності від платформи були зроблені за рахунок продуктивності. Інтерпретуючі системи, подібні до BASIC,

Perl, страждають на майже неподоланий дефіцит продуктивності. Це було враховано при створенні Java. Незважаючи на те, що Java є інтерпретованою мовою, генерація байткодів була ретельно оптимізована в такий спосіб, щоб одержуваний байткод можна було легко перекладати в машинний код, який працює з дуже високою продуктивністю. Виконуючі системи такого роду не втрачають жодних переваг переносимого коду.

Розроблена система являє собою інструмент для допомоги при постановці діагнозу. Основними функціональними характеристиками системи є:

- ведення загальної інформаційної бази з розподіленим доступом;
- авторизований доступ до даних на основі прав доступу;
- негайне відображення змін, внесених з будь-якого робочого місця;
- формування необхідних вихідних документів.

Розроблений варіант вирішення проблеми постановки діагнозу є системою підтримки прийняття рішень лікаря.

Система автоматизує наступні функції:

- Додавати інформацію про симптоми;
- Проводити пошук;
- Корегувати знайдений шаблон;
- Додавати інформацію про обстеження;
- Додавати інформацію про пацієнта;
- Затвердити діагноз;
- Друкувати документацію по проекту.

При впровадженні програмного забезпечення розробленої системи основними цілями є:

- зменшити витрати часу лікарів на пошуки потрібної інформації, що дозволить їм більш ефективно виконувати свої функції;
- знизити ризики лікарських помилок при постановці діагнозу;
- підвищення ефективності роботи лікарів;
- використання СППР дозволить працювати з реальною складністю задачі без узагальнення та спрощення.

— підвищити швидкість процесу обробки інформації і знизити витрати, пов'язані з ним;

— підвищити рівень організації і досвіду робітників в використанні інформаційних технологій, що підготує умови для наступного етапу інформатизації підприємства.

Робоча документація на систему підтримки прийняття рішень потрібна для розробки і налагодження даної системи в цілому.

Спроектована система відповідає вимогам які викладені в технічному завданні (Додаток А).

При роботі з розробленим програмним забезпеченням системи варто керуватися документом «Технічний опис» (Додаток В).

Лістинг програмного коду представлено в додатку Б.

Розроблений документ «Програма та методика випробувань» представлено в додатку Г.

4 РОЗРАХУНОК ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ ВПРОВАДЖЕННЯ СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ

4.1 Вступ

На даний момент сучасні обчислювальні засоби широко використовуються в повсякденній діяльності людини. Сьогодні обчислювальна техніка дозволяє вирішувати найрізноманітніші задачі: від звільнення людини від ручної праці до розробки складних технічних розрахунків, що не під силу виконати одній людині. Сучасні засоби програмування дозволяють вирішувати задачі, що стояли в нещодавно перед цілими колективами наукових співробітників.

Розвиток нових технологій і ріст застосування обчислювальної техніки привели до широкого застосування ЕОМ для вирішення проблем, пов'язаних з обробкою інформації, її контролем і аналізом. Сьогодні на основі засобів обчислювальної техніки розробляються і впроваджуються різні автоматизовані інформаційні системи (проектування, технологічної підготовки виробництва).

Використання даного програмного продукту приведе до значної економії, що виражається в наступних показниках:

- підвищення продуктивності праці;
- прискорення обробки інформації;
- економія часу на обробку інформації.

Найбільш важливим моментом для розробника, з економічної точки зору, є процес формування вартості програмного продукту. Очевидно, що вона являє собою дуже специфічний товар з безліччю особливостей.

Створення системи підтримки прийняття рішень вимагає одноразових витрат на її розробку, придбання необхідних технічних засобів і поточних витрат на функціонування системи. Економія від функціонування СППР визначається з урахуванням витрат на її експлуатацію. Відношення цієї економії до витрат на створення системи характеризує економічну ефективність капітальних вкладень. Економічні показники визначаються по діючим на момент розрахунку оптовим цінам, тарифам і ставкам заробітної плати.

4.2 Розрахунок витрат на створення та експлуатацію інформаційної системи

Витрати на розробку системи складаються з витрат на зарплату розробника, на амортизацію ЕОМ, на якій виконується розробка, на експлуатацію цієї ЕОМ, на засоби розробки та витрат на матеріали і комплектуючі.

Виходячи з того, що в середньому заробітна плата розробників програмного забезпечення становить 8000 грн./місяць, вартість сучасного комп'ютера, який відповідає висунутим вимогам в третьому розділі, становить 15000 грн. і вартість кіловат-годин електроенергії становить 0,90 грн, розрахуємо вартість розробки системи. Витрати на допоміжні матеріали наведені в таблиці 4.1.

Таблиця 4.1

Витрати на допоміжні матеріали

Найменування витрат	Сума, грн.
Папір	100
Заправлення картриджа до принтера	140
Компакт-диски	40
Непередбачені витрати	200
Разом матеріали і комплектуючі вироби	480

Капітальні витрати на розробку системи K_{π} розраховуються за формулою:

$$K_{\pi} = C + (Z_{\text{пр}} + Z_{\text{е}}) \cdot T + Z_{\text{д}} \quad (1)$$

де C - первісна вартість програмного продукту, грн.;

$Z_{\text{пр}}$ - заробітна плата розробника, грн.;

$Z_{\text{е}}$ – експлуатаційні витрати, грн.;

T - тривалість розробки, міс.;

$Z_{\text{д}}$ - додаткові витрати, грн.;

Заробітна плата фахівця розраховується за формулою:

$$Z_{\text{пр}} = Z_{\text{озп}} \cdot \left(1 + \frac{A_c}{100}\right) \quad (2)$$

де $Z_{\text{озп}}$ – основна заробітна плата фахівця;

A_c - відсоток відрахувань на соціальне страхування;

$$Z_{\text{пр}} = 8000 \cdot (1 + 0,34) = 10720$$

Розрахуємо Z_e при споживанні потужності 0,3 Вт, тривалості роботи на місяць рівної $21 \cdot 8 = 168$ годин і вартості кіловат-години електроенергії 0,90 грн.,

$$Z_e = 168 \cdot 0,90 \cdot 0,3 = 45,36 \text{ грн.}$$

Поточні витрати на розробку системи представленні в таблиці 4.2

Таблиця 4.2

Витрати на розробку інформаційної системи

Найменування витрат	Одиниця виміру	Кількість
Тривалість розробки	міс.	4
Заробітна платня	грн.	8000
Відрахування на соціальні заходи	грн.	2720
Витрати на допоміжні матеріали	грн.	480
Витрати на електроенергію	грн.	45,36

Відповідно до формули (1) капітальні витрати на розробку системи складають:

$$K_{\text{п}} = 15000 + (10720 + 45,36) \cdot 4 + 480 = 58541,44 \text{ грн}$$

Амортизаційні відрахування на устаткування складають 60% балансової вартості в рік:

$$A_{\text{об}} = 15000 \cdot 0,6 = 9000 \text{ грн}$$

У масштабах підприємства річні витрати на основні і допоміжні матеріали визначаються в розмірі 5% вартості основного устаткування:

$$B_{\text{м}} = 15000 \cdot 0,05 = 750 \text{ грн}$$

Річний обсяг робіт ПЕОМ у годинах визначається в такий спосіб:

$$\Phi_{\text{м}} = 253,3 \cdot T_3$$

де T_3 - це середнє місячне завантаження устаткування (близько 7 годин), 253,3 - середня кількість робочих днів у році:

$$\Phi_{\text{м}} = 253,3 \cdot 7 = 1773,1 \text{ годин}$$

Витрати на електроенергію Z_e при 1773,1 годинах роботи устаткування в рік складуть:

$$Z_e = 1773,1 \cdot 0,3 \cdot 0,90 = 478,7 \text{ грн}$$

4.3 Економічна ефективність розробки і впровадження системи

Критерієм ефективності створення і впровадження нових засобів автоматизації є очікуваний економічний ефект. Він визначається за формулою:

$$E = \Delta P - E_n \cdot K_n \quad (3)$$

де ΔP - економія від підвищення продуктивності праці;

E_n - нормативний коефіцієнт ($E_n = 0.15$);

K_n - капітальні витрати на впровадження системи.

Якщо користувач системи при економії i -го виду робіт із застосуванням програми економить T_i годин, то підвищення продуктивності праці P_i (у %) визначається за формулою:

$$P_i = \left(\frac{\Delta T_j}{F_j - \Delta T_j} \right) \cdot 100\% \quad (4)$$

де F_j - час, який планувалося затратити користувачем для виконання роботи j -виду до впровадження системи.

В таблиці 4.1 представлена інформація про вид робіт, виконуваних користувачем, та час який він витрачає.

Таблиця 4.1

Таблиця робіт користувача

Вид робіт	До автоматизації, хв. F_j	Економія часу, хв. ΔT_j	Підвищення продуктивності праці P , %
Введення інформації	40	20	100
Проведення розрахунків	90	70	350
Аналіз і вибірка даних	50	25	100
Підготовка та друк звітів	30	15	100

Економія, пов'язана з підвищенням продуктивності праці користувача P визначимо за формулою:

$$\Delta P = Z_{\pi} \cdot \sum_i \frac{P_i}{100}$$

де Z_{π} - середня заробітна плата менеджера.

Заробітна плата менеджера становить відповідно до формули (2)

$$Z_{\pi} = 5000 \cdot (1 + 0,34) = 6700 \text{ грн.}$$

Відповідно до формули (4) економія, пов'язана з підвищенням продуктивності праці користувача:

$$\Delta P = 6700 \cdot 6,5 = 43550 \text{ грн.}$$

У підсумку, відповідно до формули (4), отримуємо наступну очікувану економічну ефективність:

$$E = 43550 - 0,15 \cdot 58541,44 = 34768,78 \text{ грн.}$$

Строк окупності системи розраховується за формулою:

$$T = \frac{K_{\pi}}{\Delta P} = \frac{58541,44}{43550} \approx 1,34 \text{ року}$$

Отже строк окупності системи складає приблизно 1 рік 4 місяці.

5 ОХОРОНА ПРАЦІ

5.1 Вступ

Охорона праці – це система правил і заходів, які забезпечують безпечну роботу на даному виробництві.

Законодавство про працю містить норми і вимоги з техніки безпеки і виробничої санітарії, норми, що регулюють робочий час і час відпочинку, звільнення та переведення на іншу роботу, норми праці щодо жінок, молоді, гігієнічні норми і правила тощо.

Загальний нагляд за дотриманням норм охорони праці покладено на прокуратуру, спеціальний — на професійні спілки. Контроль за безпекою праці здійснюють також державні й відомчі спеціалізовані інспекції.

У сучасному житті комп'ютер широко застосовується в житті людини: і удома, і в офісі, і в магазині, і у виробництві, і навіть в побутовій техніці — іншими словами комп'ютери міцно увійшли до повсякденного життя людей і їх використання постійно збільшується.

В офісах комп'ютери в основному використовуються як допоміжні засоби обробки інформації, і таке введення комп'ютерних технологій принципово змінило характер праці офісних працівників і вимоги до організації і охорони праці.

Недотримання вимог безпеки приводить до того, що через деякий час роботи за комп'ютером співробітник починає відчувати певний дискомфорт: у нього виникають головні болі і різь в очах, з'являються втома і дратівливість. У деяких людей порушується сон, погіршується зір, починають хворіти руки, шия, поперек і тому подібне.

5.2 Аналіз шкідливих та небезпечних факторів

При роботі з комп'ютером шкідливими і небезпечними чинниками є:

- електростатичні поля;
- електромагнітне випромінювання;
- наявність потужних іонізуючих випромінювань;
- локальне стомлення, загальне стомлення;
- стомлюваність очей;
- небезпека поразки електричним струмом;
- пожежонебезпека.

Всі ці чинники викликані двома основними причинами: неправильною роботою за обладнанням і неправильним вибором обладнання.

Негативний вплив комп'ютера на людину є комплексним.

По-перше, комп'ютер (особливо його дисплей, або монітор, сконструйований на основі ЕПТ (електронно-променевої трубки)) є джерелом електростатичного поля; електромагнітних випромінювань в низькочастотному, наднизькочастотних та високочастотному діапазонах (2 Гц - 400 кГц); випромінювання оптичного діапазону (ультрафіолетового, інфрачервоного і видимого світла), рентгенівського випромінювання.

Вчені встановили, що випромінювання низької частоти в першу чергу негативно впливає на центральну нервову систему, викликаючи головний біль / запаморочення, нудоту, депресію, безсоння, відсутність апетиту, стрес.

Низькочастотне електромагнітне поле може з'явитися причиною шкірних захворювань (вугрі, екзема, рожевий лишай та ін), хвороб серцево-судинної системи та шлунково-кишкового тракту; воно впливає на білі кров'яні тілця, що призводить до виникнення пухлин, у тому числі і злоякісних .

Електростатичне поле великої напруженості здатне змінювати і переривати клітинне розбиток, а також в окремих випадках викликати катаракту.

Дисплеї з ЕПТ завдяки електростатичного поля інтенсивно притягають і збирають пил, а значить, можуть бути причиною появи дерматитів на шкірі обличчя, загострення бронхіальної астми, подразнення слизових оболонок

По-друге, нерухома і напружена поза.

По-третє, інтенсивна робота з клавіатурою викликає больові відчуття в ліктьових суглобах, передпліччях, зап'ястях, в кистях та пальцях рук.

Відповідно до законодавчих актів площа робочого місця користувача ПК повинна складати не менше 4,5м². У приміщеннях повинні проводитися щоденне вологе прибирання і систематичне провітрювання після кожної години роботи. Устаткування, що виробляє багато шуму (друкуючі пристрої, сканери, сервери і тому подібні), рівні шуму якого перевищують нормативні, повинне розміщуватися поза робочими місцями співробітників.

Робочі столи слід розміщувати так, щоб монітори були орієнтовані бічною стороною до світлових отворів, щоб природне світло падало переважно зліва.

При розміщенні робочих місць відстань між робочими столами має бути не менше 2,0 м-коду, а відстань між бічними поверхнями відеомоніторів — не менше 1,2 м. Робочі місця співробітників, що виконують творчу роботу і що вимагає значної розумової напруги або високої концентрації уваги, рекомендується ізолювати один від одного перегородками заввишки від 1,5 м.

Конструкція робочого столу повинна забезпечувати оптимальне розміщення на робочій поверхні використовуваного устаткування. Висота робочої поверхні столу повинна складати 725 мм, робоча поверхня столу повинна мати ширину 800..1400 мм і глибину 800..1000 мм. Робочий стіл повинен мати простір для ніг заввишки не менше 600 мм, шириною — не менше 500 мм, завглибшки на рівні колін — не менше 450 мм і на рівні витягнутих ніг — не менше 650 мм.

Конструкція робочого стільця або крісла повинна забезпечувати підтримку раціональної робочої пози працівника і дозволяти змінювати позу з метою зниження статичної напруги м'язів шийно-плечової області і спини. Робочий стілець або крісло мають бути підйнятно-поворотними, регульованими по висоті і кутам нахилу сидіння і спинки, а також відстані спинки від переднього краю сидіння, при цьому регулювання кожного параметра має бути незалежним, легко здійснюваним і мати надійну фіксацію.

Клавіатуру слід розташовувати на поверхні столу на відстані 100..300 мм від краю, зверненого до користувача, або на спеціальній поверхні, відокремленій від основної.

Екран відеомонітора повинен знаходитися від очей користувача на відстані 600..700 мм, але не ближче 500.

5.3 Розрахунок системи штучного освітлення приміщення відділу інформаційних технологій

У відділі інформаційних технологій використовується загальне штучне освітлення, тобто освітлення приміщень штучним світлом за допомогою електричних ламп, яке використовується при недостатньому природному освітленню.

Необхідно зробити розрахунок загального освітлення виробничого приміщення. Довжина $A = 40$ м, ширина $B = 14$ м, висота $H = 3,5$ м. Висота виробничої поверхні $h_p = 1$ м. Для освітлення приймаємо світильники типу НСП 07. Мінімальна освітленість лампи розжарювання за нормами $E_{\min} = 100$ лк. Коефіцієнт відображення стелі $S_n = 70\%$, стін $S_c = 50\%$, робочої поверхні $S_p = 30\%$. Напруга мережі 220 В.

Розрахунок здійснюємо за методом коефіцієнта використання світлового потоку.

Відстань від стелі до робочої поверхні

$$H_o = H - h_p = 3,5 - 1 = 2,5 \text{ м.}$$

Відстань від стелі до світильника

$$h_c = 0,2 \cdot H_o = 0,2 \cdot 2,5 = 0,5 \text{ м.}$$

Висота підвішування світильника над освітлювальною поверхнею

$$h = H_o - h_c = 2,5 - 0,5 = 2 \text{ м.}$$

Висота підвішування світильника над підлогою

$$H_n = h + h_p = 2 + 1 = 3 \text{ м.}$$

Для досягнення найбільшої рівномірності освітлення приймаємо відношення $L/h = 1,5$. Тоді відстань між центром світильників складає

$$L = 1,5 \cdot h = 1,5 \cdot 2 = 3 \text{ м.}$$

Необхідна кількість світильників

$$N = S/L^2 = (40 \cdot 14) / 3^2 = 62,22 \text{ світ.}$$

Приймаємо 63 світильники - 7 рядів по 9 світильників (рисунок 5.1).



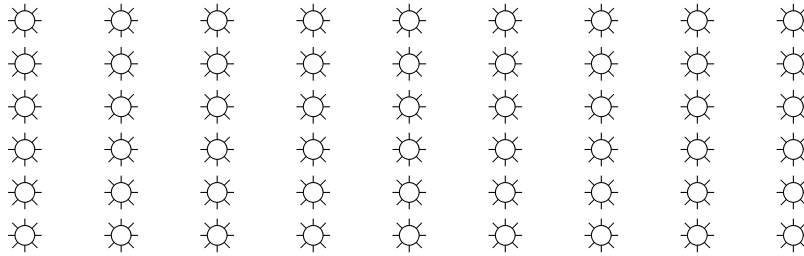


Рисунок 5.1 - Розташування світильників у виробничому приміщенні

Індекс приміщення дорівнює

$$i = A \cdot B / (h \cdot (A + B)) = (40 \cdot 14) / (2 \cdot (40 + 14)) = 560 / 108 = 5,185$$

При $i = 5,185$; $S_n = 70\%$, $S_c = 50\%$, $S_p = 30\%$ для світильника типу НСП 07 коефіцієнт використання світлового потоку $\eta = 0,9$.

Світловий потік однієї лампи

$$\Phi = (E_{min} S K_3 Z) / (N \eta) = (100 \cdot 560 \cdot 1,3 \cdot 1,15) / (63 \cdot 0,9) = 1476,54 \text{ лм},$$

де $K_3 = 1,3$ за таблицею значень коефіцієнту запасу; $Z = 1,15$ для ламп розжарювання.

За знайденим світловим потоком з таблиці параметрів ламп розжарювання загального призначення вибираємо лампу потужністю 100 Вт, що має світловий потік $\Phi_{л} = 1450$ лм, найбільш близький до розрахункового.

При цьому фактична освітленість

$$E_{\phi} = E_{min} \cdot (\Phi_{л} / \Phi_p) = 100 \cdot (1450 / 1476,54) = 98,2 \text{ лк}.$$

Загальна потужність освітлювальної установки

$$P_3 = P_{л} \cdot N = 100 \cdot 63 = 6300 \text{ Вт} = 6,3 \text{ кВт}.$$

Отже для оптимальної роботи в приміщенні необхідно встановити 7 рядів по 9 світильників. Загальна потужність усіх світильників складе 6,3 кВт.

5.4 Розробка заходів щодо зменшення впливу шкідливих та небезпечних факторів при роботі з персональним комп'ютером

5.4.1 Заходи щодо зменшення впливу

Заходи щодо усунення небезпеки поразки електричним струмом зводяться до правильного розміщення устаткування та електричних кабелів. Інші заходи

щодо забезпечення електробезпеки, збігаються з загальними заходами пожежогото-та електробезпеки.

В якості профілактичних заходів для забезпечення пожежної безпеки слід використовувати скриту електромережу, надійні розетки з пожежобезпечних матеріалів, силові мережі живлення устаткування виконувати кабелями, розрахованими на підключення в 3-5 разів більшого навантаження, включати й виключати живлення обладнання за допомогою штатних вимикачів. Треба регулярно робити очистку внутрішніх частин комп'ютерів, іншого устаткування від пилу, розташовувати комп'ютери на окремих не спалюваних столах. Для запобігання іскріння необхідно рідше встромляти і виймати штепсельні вилки з розеток.

Екран дисплея повинен бути розташованим перпендикулярно до напрямку погляду. Якщо він розташований під кутом, то стає причиною сутулості. Відстань від дисплея до очей повинна трохи перевищувати звичну відстань між книгою та очима. Перед екраном монітора, особливо старих типів, повинен бути спеціальний захисний екран. При його відсутності треба сидіти на відстані витягнутої руки від монітора. Ще одним моментом, який стосується зору, є необхідність створення неоднорідного поля зору. Для цього можна розвісити на поверхнях (стінах) плакати та картини, виконані у спокійних тонах. Наприклад, пейзажі.

Важливою є форма спинки крісла, яка повинна повторювати форму спини. Висота крісла повинна бути такою, щоб користувач не відчував тиску на куприк або стегна. Крісло бажано обладнати бильцями. Його потрібно встановити так, щоб не треба було тягтися до клавіатури. Періодично користувачу необхідно рухатися, вчасно змінювати положення тіла і робити перерви у роботі.

При напруженій роботі за комп'ютером щогодини необхідно робити перерву на 15 хвилин через кожну годину і треба займатися іншою справою. Декілька разів на годину бажано виконувати серію легких вправ для розслаблення.

5.4.2 Техніка безпеки при роботі на комп'ютері

Загальні положення:

1) При виконанні робіт на персональних комп'ютерах (ПК) необхідно дотримуватись вимог загальної та даної інструкції з охорони праці.

2) До самостійної роботи на комп'ютерах допускаються особи, які пройшли медичний огляд, навчання з професії, ввідний інструктаж з охорони праці і первинний інструктаж з охорони праці на робочому місці. В подальшому вони проходять повторні інструктажі з охорони праці на робочому місці один раз на півріччя.

3) При роботі на комп'ютерах, до складу яких входять відеодисплейні термінали (ВДТ), на працівників можуть діяти наступні шкідливі та погрожуючі життю і здоров'ю виробничі фактори:

а) фізичні :

- підвищений рівень шуму (від вентилятора мікропроцесора та плат);
- високий рівень напруги в електричній мережі, торкання до якої може викликати нещасний випадок;
- підвищений рівень статичної електрики;
- підвищений рівень електромагнітного випромінювання;
- підвищений рівень напруженості електричного поля;
- пряма і відбита від екрана блискіть;
- несприятливе розподілення яскравості об'єктів в полі зору.

б) психофізіологічні:

- фізичні перевантаження статичної та динамічної дії;
- нервово-психічні перевантаження (розумове перенапруження, перенапруження зорових аналізаторів, монотонність праці, емоційні перевантаження).

4) Основним обладнанням робочого місця користувача ПК є: ВДТ, системний блок, клавіатура, друкуючий пристрій (принтер), зовнішні пристрої пам'яті, сканер, "миша", блок безперервного живлення, робочий стіл, стілець та інше; допоміжним – шафи, полиці, сейф та ін.

Робочі місця з ВДТ повинні бути розташовані на відстані не менше 1,5 м від стіни з вікнами, від інших стін – на відстані 1 м, між собою – на відстані не менше 1,5 м. Відносно вікон робочі місця доцільно розташувати таким чином, щоб природне світло падало на нього збоку, переважно зліва.

5) Робочі місця, обладнані ВДТ, слід розташовувати таким чином, щоб запобігти попаданню в очі прямого світла. Джерела освітлення слід розташовувати з обох боків екрану, паралельно напрямку зору.

Для запобігання світлових відблисків від екрана, клавіатури в напрямку очей користувача від освітлювачів загального призначення або сонячних променів необхідно застосовувати антивідблискові сітки, спеціальні фільтри, захисні козирки, на вікнах регульовані жалюзі.

Фільтри із металевої або нейлонової сітки використовувати не рекомендується тому, що сітка спотворює зображення внаслідок інтерференції світла. Найкраща якість зображення забезпечується скляними поляризаційними фільтрами – вони усувають практично усі відблиски і забезпечують чітке та контрастне зображення.

6) При роботі з текстовою інформацією (в режимі введення даних, редагування тексту і читання з екрану ВДТ) найбільш фізіологічним є зображення чорних знаків на світлому (білому) фоні.

7) Розташовувати ВДТ на робочому місці слід так, щоб поверхня екрану знаходилась в центрі поля зору на відстані 400-700 мм від очей користувача. Пропонується розташовувати елементи робочого місця таким чином, щоб підтримувалась однакова відстань від очей користувача до екрана, клавіатури, пупітра.

8) Робота комп'ютера супроводжується електромагнітним випромінюванням різних частот і рівнів, інтенсивність котрого зменшується пропорційно квадрату відстані до екрану. Оптимальною для працюючого за ПК є відстань 50 см від екрану ВДТ.

9) Раціональною робочою позою працюючого за ПК вважається таке розташування тіла, при якому ступні працівника розташовані горизонтально на підлозі або на підставці для ніг, стегна зорієнтовані в горизонтальній площині, верхні кінцівки рук – вертикально, кут ліктьового суглоба коливається в межах 70-90 град., зап'ястя зігнуті під кутом, не перевищує 20 град., нахил голови – в межах 15-20 град.

10) Для нейтралізації зарядів статичної електрики в приміщенні, де виконуються роботи на комп'ютерах, в тому числі на лазерних та світлодіодних

принтерах, пропонується збільшувати вологість повітря за допомогою кімнатних зволожувачів. Не рекомендується носити одяг з синтетичних матеріалів.

Вимоги безпеки перед початком роботи:

1) Перевірити надійність встановлення апаратури на робочому столі. ВДТ не повинен стояти на краю столу. Повернути ВДТ так, щоб було зручно дивитись на екран – під прямим кутом (а не збоку) і трохи згори вниз, а екран повинен бути нахиленим під невеликим кутом (нижній край ближче до оператора).

2) Перевірити загальний стан апаратури, справність проводки електромережі, з'єднувальних шнурів, штепсельних вилок та розеток, заземлення захисного екрана.

4) Відрегулювати освітленість робочого місця.

5) Відрегулювати і зафіксувати висоту стільця (крісла), зручний для користувача нахил його спинки.

6) Під'єднати до системного блока необхідну апаратуру (принтер, сканер і т.п.). Всі кабелі, що з'єднують системний блок з іншими пристроями слід вставляти і виймати тільки при вимкненому електричному живленні.

7) Увімкнути апаратуру ПК вмикачами на корпусах в такій послідовності: стабілізатор (або блок безперервного живлення), ВДТ, системний блок, принтер (якщо передбачається друкування).

8) Відрегулювати яскравість світіння екрану ВДТ, фокусировку, контрастність. Не слід робити яскравість екрану занадто великою, тому що це втомлює очі (і зменшує термін роботи електронно-променевої трубки).

Рекомендується:

- яскравість світіння екрану – не менше 100 кд/ кв.м (кандел на кв.м);
- відношення яскравості екрану ВДТ до яскравості оточуючих його поверхонь в робочій зоні – не більше 3:1;
- мінімальний розмір точки світіння – не більше 0,4 мм для монохромного ВДТ і не більше 0,6 мм для кольорового;
- контрастність зображення знака – не менше 0,8.

9) При появі несправностей комп'ютерної техніки повідомити керівника.

Вимоги безпеки при закінчення роботи на ПК:

- 1) Закінчити і зберегти в пам'яті ПК файли, які знаходились у роботі. Виконати всі дії для коректного завершення роботи в оперативній системі.
- 2) Вимкнути принтер та інші периферійні пристрої, вимкнути ВДТ і системний блок. Вимкнути стабілізатор (або блок безперервного живлення при його наявності). Штепсельні вилки витягнути з розеток. Накрити клавіатуру кришкою для попередження попадання в неї пилу.
- 3) Навести порядок на робочому місці.
- 4) Вимкнути освітлення та електроприлади.

Вимоги безпеки при аварійних ситуаціях:

- 1) При раптовому припиненні подачі електричної енергії вимкнути всі пристрої ПК в такій послідовності: периферійні пристрої, ВДТ, системний блок, стабілізатор (або блок безперервного живлення). Витягнути вилки з розеток.
- 2) При наявності ознак горіння (дим, запах горілого) необхідно вимкнути всі пристрої ПК. Знайти місце загоряння і виконати всі можливі заходи для його ліквідації, попередивши терміново про це керівництво.
- 3) У випадку виникнення пожежі негайно попередити про це пожежну частину та керівництво, виконати усі можливі заходи по евакуації людей з приміщення і розпочати гасіння пожежі первинними засобами пожежогасіння.

6 ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

6.1 Вступ

Охорона навколишнього середовища — система заходів щодо раціонального використання природних ресурсів, збереження особливо цінних та унікальних природних комплексів і забезпечення екологічної безпеки, також це сукупність державних, адміністративних, правових, економічних, політичних і суспільних заходів, спрямованих на раціональне використання, відтворення і збереження природних ресурсів землі, обмеження негативного впливу людської діяльності на навколишнє середовище.

Відносини у галузі охорони навколишнього середовища регулюються Законом України «Про охорону навколишнього природного середовища», а також земельним, водним, лісовим законодавством, законодавством про надра, про охорону атмосферного повітря, про охорону і використання рослинного і тваринного світу та іншим спеціальним законодавством.

Метою охорони навколишнього середовища є протидія негативним змінам у навколишньому середовищі, які мали місце в минулому, відбуваються зараз або можуть бути.

Природоохоронні заходи мають на меті поліпшення стану навколишнього природного середовища або створення умов для цього. Ознаками природоохоронних заходів є:

- підвищення екологічності продукції, що випускається;
- скорочення споживання природних ресурсів на одиницю продукції, що випускається та здійснення господарської діяльності;
- зниження забруднення природних комплексів викидами, стоками, відходами, випромінюваннями;
- зниження концентрації шкідливих речовин у викидах, стоках, відходах;
- поліпшення стану середовища існування людей.

Актуальність охорони довкілля, що перетворилася в глобальну проблему, пов'язана зі зростаючим антропогенним впливом. Це зумовлено демографічним

вибухом, урбанізацією, що прискорюється розвитком комунікацій, забрудненням навколишнього середовища відходами, надмірним навантаженням на орні землі, пасовища, ліси, водойми.

6.2 Забруднення навколишнього середовища

При виробництві різних композиційних матеріалів відбувається викид великої кількості техногенних відходів в навколишнє середовище, що суттєво забруднює атмосферу.

Оскільки на підприємствах для виробництва КМ використовуються різноманітні технології, то і в довкілля потрапляють різні забруднюючі речовини.

Основними джерелами утворення і викидів в атмосферу забруднюючих речовин є наступні технологічні процеси: підготовка сировини (збагачування, агломерація), виробництво коксу, доменне виробництво, виробництво сталі, виробництво прокату.

Наприклад бетонні виробництва потребують близько 20 технологічних процесів (розмелювання, висушування, подрібнення, нагрівання в печах, охолодження в баштах, транспортування на стрічкових конвеєрах, транспортування готового продукту тощо), що супроводжуються забрудненням повітря, причому кожен з них має особливості і створює власні технічні проблеми. Найбільш складними технологічними блоками є хімічні підприємства. На одному заводі ряд виробничих процесів може спричинити викиди різних забруднюючих речовин, включно з газоподібними (наприклад, при виробництві азотної, сірчаної кислот, віскози та добрив, а також теплової енергії в котельнях).

Для аглофабрики кількість агломераційних газів, що викидаються в атмосферу, становить - 1,83-106 м³ / год, їх запиленість 8-10 г / м³, зміст CO₂ - 4,2%, CO - 0,6-0,9%, SO₂ - 0,05%. Під час агломерації руди відбувається вигорання сірки з тиритів. Сульфідні руди містять до 10% сірки, а після агломерації її залишається лише 0,2 - 0,8%. Викид сірчистого газу при

агломерації може бути у розмірі 190 кг. на 1 т. руди, тобто одна стрічкова машина дає близько 700 т. сірчистого газу за добу.

У доменному виробництві мають місце викиди в атмосферу продуктів згоряння з доменної печі (колошникових газів) у процесах завантаження шихти, випуску чавуну, охолодження шлаку. Також разом з доменним газом в атмосферу в невеликих кількостях викидаються також сполуки миш'яку, фосфору, сурми, свинцю, пари ртуті і рідких металів, ціанистий водень і смолисті речовини.

У загальному балансі пилогазових викидів сталеплавільних агрегатів на частку виробництва сталі припадає 75% пилу, 85% оксидів азоту, 90% діоксиду сірки. Запиленість димів газів становить 0,5-2 г / м³.

Значну роль у забрудненні атмосфери грають викиди мартенівських і конверторних сталеплавильних цехів. При веденні мартенівського процесу пил утворюється з металевої шахти (сталевий брукхт при її окислюванні, зі шлаку, руди, вапняку, що йдуть на окислювання домішок шихти, і з доломіту, що застосовується для заправлення підлоги печі). У період кипіння сталі виділяються також пари окисів шлаку і металу, газу. Розміри часток пилу не перевищує 3 мкм. Переважна частина пороши мартенівських печей складається з триокису заліза (67%) і триокису алюмінію (8,7%). При безкисневому процесі на 1 т. мартенівської сталі виділяється 3000 - 4000 м³ газів з концентрацією пилу в середньому 0,5 г/м³. У період подачі кисню в зону розплавленого металу пилоутворення багаторазово збільшується, досягаючи 15-52 г/м³. Крім того, плавлення сталі супроводжується вигоранням деякої кількості вуглецю і сірки, у зв'язку з чим у газах мартенівських печей, що відходять при кисневому дутті, міститься до 60 кг. окису вуглецю і до 3 кг. сірчистого газу в розрахунку на 1 т. виданої сталі. Головною особливістю конверторного процесу є одержання сталі з рідкого чавуну без застосування палива. Виплавка сталі за таким принципом здійснюється в конвертерах ємністю 50,100, 250 і більше тонн шляхом продування рідкого чавуну киснем. Це забезпечує вигорання небажаних домішок, наприклад, марганцю, фосфору і вуглецю, що містяться у чавуні. Процес одержання конверторної сталі носить циклічний характер і при кисневому дутті триває 25-30 хвилин. Димові газу, що утворюються,

складаються з часток окисів кремнію, марганцю і фосфору. У складі диму міститься значна кількість окису вуглецю - до 80%. Концентрація пилу в газах що відходять, складає приблизно 15 г/м³. Основна маса пилу -90%, складається з часток розміром 0,2 -1 мкм.

У процесі виробництва коксу мають місце викиди в атмосферу коксового газу. Коксохімічні підприємства забруднюють атмосферне повітря пилом і сумішшю летких сполук. Процес коксування полягає в термічній, без доступу повітря, обробці кам'яновугільної шихти при температурі 900 - 1100вС. При коксуванні 1т. вугілля утворюється 300-320 м³ коксового газу. У його сполуки входять: водень - 50-62% (об'ємних), метан -20-34%, окис вуглецю - 4,5-4,7%, вуглекислий газ -1,8-4%, азот - 5-10%, вуглеводні - 2-2,6% і кисень -0,2-0,5%.

Основна маса коксового газу вловлюється і направляється на хімічну переробку й утилізацію, близько 6% газу надходить в атмосферу внаслідок втрат під час завантаження і вивантаження печей і негерметичності апаратури. У деяких випадках, наприклад, при порушенні режиму роботи батарей коксування чи допоміжних відділень утилізації, в атмосферу викидаються значні кількості неочищеного коксового газу.

6.3 Розробка заходів щодо зменшення забруднення навколишнього середовища

Охорона навколишнього середовища на підприємстві характеризується комплексом вжитих заходів, які спрямовані на попередження негативного впливу людської діяльності підприємства на навколишню природу, що забезпечує сприятливі та безпечні умови людської життєдіяльності. Враховуючи стрімкий розвиток науково-технічного прогресу, перед людством постала складна задача — охорона найважливіших складових навколишнього середовища (земля, вода, повітря), схильних сильному забрудненні техногенними відходами і викидами, що призводить до окислення ґрунту і води, руйнування озонового шару землі та кліматичним змінам . Промислова політика всього світу привела до таких незворотних і суттєвих змін в навколишньому середовищі, що це питання (охорона навколишнього середовища на

підприємстві) став загальносвітовою проблемою і примусив державні апарати розробити довгострокову екологічну політику зі створення внутрішньодержавного контролю.

Основними умовами для поліпшення екології в країні є: раціональне використання, охорона і трата запасів природного резерву, забезпечення безпеки екології та протирадіаційні заходи, підвищення і формування екологічного мислення у населення, а також контроль над екологією в промисловості. Охорона навколишнього середовища на підприємстві визначила ряд заходів для зниження рівня забруднень, що виробляється підприємствами:

- виявлення, оцінка, постійний контроль та обмеження викиду шкідливих елементів в атмосферу, а також створення технологій і техніки, які охороняють і зберігають природу і її ресурси.
- розробка правових законів, спрямованих на охоронні заходи навколишнього середовища та матеріальне стимулювання виконаних вимог і профілактики комплексу природоохоронних заходів.
- профілактика екологічної обстановки шляхом виділення спеціально відведених територій (зон).

Суттєвого зменшення забруднення навколишньої середовища підприємствами можна досягти впровадженням таких заходів:

- 1) Реалізація засад ресурсозбереження в усьому ланцюзі суспільного виробництва на шляху його інтенсифікації і зниження ресурсомісткості (водо-, земле-, метало-, енерго-);
- 2) Впровадження безвідходних технологій, що означає (і має своїм наслідком): комплексне використання сировини і енергоносіїв, створення замкнених газо- і водооборотних систем, застосування принципово нових підходів до вилучення, збагачення, перероблення сировини і матеріалів – біотехнологій, геотехнологій.

При вирішенні проблеми безвідходності виробництва слід мати на увазі дві сторони єдиного процесу. Перше – це найбільш раціональних видобуток та повне використання ресурсів і як наслідок зменшення утворення відходів. Друге – це розширення використання відходів, що утворюються. Ці шляхи не виключають, а взаємно доповнюють один одного.

Поняття безвідходних технологій дещо умовним, оскільки повної безвідходності досягти практично не можливо. Більш коректним буде говорити про маловідходні технології. При цьому мається на увазі можливість створення технологічних систем, вплив яких на природу не буде перевищувати її відновлювального потенціалу.

В основі концепції безвідходних технологій лягли три основні положення, а саме:

- створення максимально замкнених систем, організованих за аналогією з природними екосистемами;
- раціональне використання всіх компонентів сировини;
- неминучі впливи на навколишнє середовище не повинні порушувати його функціонування.

ВИСНОВКИ

У представленій роботі вирішено актуальну науково-практичну задачу підвищення ефективності процесу діагностування очних захворювань за рахунок розробки інформаційної технології у вигляді системи підтримки прийняття рішень в умовах невизначеності.

У результаті виконання магістерської роботи було вирішено наступні задачі:

- Проведений аналіз процесу постановки діагнозу.
- Проведений аналіз методів постановки діагнозу.
- Досліджені клінічні системи підтримки прийняття рішень.
- Запропоновано модель постановки діагнозу лікарем, яка реалізована за методами СППР.
- Розроблено математичні моделі, які можуть використатися для розробки СППР для постановки діагнозу.
- Було розроблено проект СППР для проектування ТП.
- Розроблена робоча документація по спроектованій системі.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

- 1 **Eike-Henner, W. K.** E-Healthanditschallenges [Text] / W. K. Eike-Henner // Healthcare IT Management. – 2010. – Vol. 2, Issue 5. – P. 32–33.
- 2 **Бабук, І.М.** Економіка підприємства: навч. посібник для студентів технічних спеціальностей [Текст] / І.М.Бабук. - Мінськ: "ІВЦ Мінфіну", 2006.
- 3 **Бейко, І. В.** Проектування СППР на основі нечіткої логіки. Навчально-методичний посібник / **І. В. Бейко, В. В. Акіменко, Ю. В. Загородній.**
- 4 **Буч, Г.** Мова UML. Посібник користувача [Текст] / Г.Буч, Дж. Рамбо, А. Джекобсон. – К.: ДМК прес, 2007. – 496 с.
- 5 **Джигирей, В. С.** Екологія та охорона навколишнього природного середовища: Навчальний посібник [Текст] / В. С. Джигирей. - 3-є вид., випр. і доп. - К.: Т-во "Знання", КОО, 2004. - 309 с.
- 6 **Долгополов, И. Н.** Прийняття рішень у задачах клінічної діагностики в умовах невизначеності [Текст] / И. Н. Долгополов // Системи підтримки прийняття рішень. Теорія і практика. – К., 2008. – с. 133-135.
- 7 Екологічний менеджмент: Навчальний посібник [Текст] / В. Ф. Семенов, О. Л. Михайлюк, Т. П. Галушкіна, Г. В. Крусір та ін.; За ред. В. Ф. Семенова, О. Л. Михайлюк ; М-во освіти і науки України, ОДЕУ. - К. : Центр навчальної літератури, 2004. - 407 с.
- 8 Закон України про охорону навколишнього природного середовища [Текст] : затверджено і введено в дію Постановою Верховної Ради України №1268-ХІІ від 26 червня 1991, чинний, поточна редакція від 18 листопада 2012.
- 9 Інструкція з охорони праці при роботі на комп'ютерах [Текст] : затверджено і введено в дію наказом Мінінформа України №75 від 05 червня 1996 р.
- 10 **Коваленко, І. І.** Методи та засоби підтримки прийняття рішень: Навчальний посібник [Текст] / І. І. Коваленко, А. П. Гожий. – Миколаїв, 2005. – 104 с.
- 11 **Коваленко, І. І.** Методи прийняття рішень: Навчальний посібник [Текст] / І. І. Коваленко, Т. А. Фарионова, С. Б. Приходько. – Миколаїв, НУК, 2009. – 180 с.

12 **Коваленко, І. І.** Сучасні методи статистичного аналізу даних: Навчальний посібник [Текст] / І. І. Коваленко, С. Б. Приходько, Л. О. Латанська. – Миколаїв: НУК, 2011. – 192 с.

13 Комп'ютеризовані інтегровані виробництва: основи організації та приклади використання: Навч. посібник для вузів [Текст] / К. В. Кошкін, А. С. Суслов, С. М. Хальнов, В. В. Шишканов. – Миколаїв: НУК, 2006 – 180 с.

14 **Коробин, В. И.** Екологія: Підручник для вузів [Текст] / В. И. Коробин, Л. В. Передельський. – Ростовн / Д: вид-во «Фенікс», 2000. – 576 с.

15 **Кривошеєв, К. В.** Моделі, методи та інформаційна технологія підтримки прийняття рішень при діагностуванні очних захворювань [Текст] / К. В. Кривошеєв. - «Молодий вчений» № 11 (26) Ч. 1 , 2015. – с. 28-32.

16 **Мельникова, Н. І.** Аналітичний огляд засобів програмного забезпечення в медичній галузі [Текст] / Н. І. Мельникова, Н. Б. Шаховська. – 2010. – с. 146-153.

17 **Тубальцев, А. М.** Виробниче освітлення та його розрахунок: Навчальний посібник [Текст] / А. М. Тубальцев. – Миколаїв: УДМТУ, 2001. – 84 с.

ДОДАТОК А – Технічне завдання

1 Загальні положення

Назва інформаційної системи: «СППР в медичній діагностиці».

Створюваний програмний продукт являє собою систему підтримки прийняття рішень для постановки діагнозу лікарем. Дана автоматизована система забезпечує ведення та управління базою даних симптомів, даних аналізів, діагнозів, а також забезпечує генерацію діагнозів.

Підстави для розробки: Наказ №1307-уч від 08.10.2024 «Про затвердження тем магістерських робіт».

2 Призначення та мета створення системи

2.1 Призначення

Дана система призначена для підтримки прийняття рішень при постановці діагнозу лікарем.

2.2 Мета створення

Метою створення даної системи є підвищення ефективності роботи лікарів, усунення можливих лікарських помилок, втрат інформації та економія часу. Експлуатаційне призначення: скорочення термінів постановки діагнозів, робота з реальною складністю задачі без узагальнення та спрощення.

3 Характеристика об'єкту автоматизації

Об'єктом автоматизації даного проекту є задача постановки діагнозу лікарем. Під автоматизацію підпадає функції ведення бази даних симптомів, даних аналізів, діагнозів, а також забезпечує генерацію діагнозів. Дані процеси описані в розділі 1 та 2 даної роботи.

4 Цілі та завдання розробки системи

Ціллю даної СППР є:

- збільшення кількості користувачів, що працюють із системою;
- зведення до мінімуму суб'єктивних помилок;
- підвищення достовірності даних;
- створення єдиної інформаційної системи;
- підвищення надійності функціонування.

Створювана СППР повинна виконувати наступні функції:

- Ведення бази даних симптомів;
- Ведення бази даних аналізів;
- Генерація альтернатив.

5 Вимоги до системи

5.1 Вимоги до функціональних характеристик

Програмне забезпечення повинно передбачати наступні функції:

- Додавати інформацію про симптоми;
- Проводити пошук;
- Корегувати знайдений шаблон;
- Додавати інформацію про обстеження;
- Додавати інформацію про пацієнта;

- Затвердити діагноз;
- Друкувати документацію по проекту.

5.2 Вимоги до надійності

Для коректного функціонування програмного додатку необхідно забезпечувати:

- Безперебійне живлення технічних засобів;
- Використання ліцензійного програмного забезпечення;
- Випробування програмних засобів на наявність комп'ютерних вірусів.

Необхідно слідкувати за коректністю введення даних. Якщо обов'язкові поля не заповнені програма повинна видати про це повідомлення і не продовжувати подальшу роботу то того часу як дані поля не будуть заповнені. Також якщо введена інформація не відповідає типу поля програма повинна видати про це попередження. При введенні помилкової інформації програма повинна забезпечувати її редагування.

5.3 Вимоги до експлуатації

Треба враховувати що відділ технологів може розташовуватись в цехових приміщеннях. Комп'ютер технолога повинний бути спеціально призначений для експлуатації в жорстких цехових умовах. Ця спецтехніка має найвищу ступінь захисту від пилу і відповідає рівню IP65 по стандарту ISO9001, що забезпечує пило-ізоляцію та безпроблемний контакт з водою.

Від користувача вимагається знання інструкції користувача, та загальні вміння користуватися ПК.

5.4 Вимоги до складу та параметрів технічних засобів

- серверна платформа CPU AMD ATHLON 64 3800+(2.4 ГГц), RAM 1024

Мб, HDD 200 Гб і серверне програмне забезпечення (MS Windows 2003 Server);

— клієнтські платформи : CPU AMD Sempron 3000+(1,6 ГГц), RAM 512 Мб, HDD 80 Гб та клієнтське програмне забезпечення (MS Windows 7);

— мережне середовище (100 Mbit);

Апаратне забезпечення повинне максимально повно охоплювати існуючий парк комп'ютерної техніки на підприємстві.

5.5 Вимоги до маркування та пакування

Програмний додаток повинен мати маркування з зазначенням торгового знаку компанії-розробника, найменування, номеру версії та дати виготовлення.

Маркування повинне бути нанесене у вигляді наліпки, виробленій поліграфічним засобом.

Пакування програмного виробу повинне здійснюватися в пакувальну тару виробника.

5.6 Вимоги до транспортування та зберігання

Допускається транспортування програмного виробу в транспортній тарі усіма транспортними засобами. При зберіганні та транспортуванні повинен здійснюватись захист від попадання пилу та вологи.

6 Вимоги до програмної документації

Програмна документація повинна містити:

- Технічне завдання;
- Технічний опис;
- Програму та методики випробувань;
- Інструкцію користувача;

7 Стадії та етапи розробки

Роботи по створенню системи включають в себе наступні етапи та стадії розробки:

- Перед проектне дослідження:
- Аналіз предметної галузі;
- Оформлення вимог користувача до системи;
- Розробка концепції системи:
- Розробка варіантів концепцій системи та їх функціональний аналіз;
- Вибір варіанту концепції системи.
- Ескізне проектування:
- Прийняття загальносистемних рішень;
- Розробка попередніх рішень по вибору варіантів інформаційного, технічного та програмного забезпечення системи.
- Технічне проектування:
- Прийняття проектних рішень по системі та її частинам;
- Розробка технічної документації по проекту.
- Ввід в дію:
- Виготовлення та налагодження компонентів системи;
- Розробка акту передачі.

8 Порядок прийому та контролю

Порядок контролю і здачі даного програмного продукту виконується представником замовника у присутності представника розробника згідно програмі і методиці випробування.

Для контролю і здачі даного програмного продукту повинна бути представлена розроблена АС, методика випробувань і інструкція користувача.

За результатами задачі складається акт, який підписується представником замовника і представником розробника, і затверджується керівниками організації-замовника і організації-розробника.

ДОДАТОК Б – Текст програми

```
package spprdl;

public class Main {

    mainForm mf;

    public static void main(String[] args) {
        Main prog = new Main();
        prog.run();
    }

    public void run() {
        mf = new mainForm();
        mf.setVisible(true);
    }
}

package spprdl;

public class mainForm extends javax.swing.JFrame {

    projectForm pf;

    public mainForm() {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        jPanel2 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jComboBox1 = new javax.swing.JComboBox();
        jTextField1 = new javax.swing.JTextField();
        jButton1 = new javax.swing.JButton();
        jButton2 = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE
);

        jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.LineBorder(new java.awt.Color(0, 0, 0), 1,
true), " СППР для постановки діагнозу лікарем "));

        jPanel2.setBorder(javax.swing.BorderFactory.createLineBorder(new
java.awt.Color(0, 0, 0)));
    }
}
```

```

jLabel1.setText("Загальна інформація:");

jLabel2.setText("Назва проекту:");

jLabel3.setText("Автор:");

javax.swing.GroupLayout jPanel2Layout = new
javax.swing.GroupLayout(jPanel2);
jPanel2.setLayout(jPanel2Layout);
jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel2Layout.createSequentialGroup()
        .addGap(10, 10, 10)
        .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
            .addComponent(jLabel1)

            .addGroup(jPanel2Layout.createSequentialGroup()
                .addGap(10, 10, 10)
                .addComponent(jLabel2)

                .addGap(10, 10, 10)
                .addComponent(jTextField1,
javax.swing.GroupLayout.PREFERRED_SIZE, 162,
javax.swing.GroupLayout.PREFERRED_SIZE))

            .addGroup(jPanel2Layout.createSequentialGroup()
                .addGap(10, 10, 10)
                .addComponent(jLabel3)

                .addGap(10, 10, 10)
                .addComponent(jComboBox1, 0,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
        .addContainerGap(10, Short.MAX_VALUE))
    );
jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanel2Layout.createSequentialGroup()
        .addGap(10, 10, 10)
        .addComponent(jLabel1)

        .addGap(10, 10, 10)
        .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
            .addComponent(jLabel2)
            .addComponent(jTextField1,
javax.swing.GroupLayout.PREFERRED_SIZE,

```

```

javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayou
t.Alignment.BASELINE)
        .addComponent(jLabel3)
        .addComponent(jComboBox1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
    };

    jButton1.setText("X");
    jButton1.addActionListener(new
java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent
evt) {
            jButton1ActionPerformed(evt);
        }
    });

    jButton2.setText("Початок діагностики");
    jButton2.addActionListener(new
java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent
evt) {
            jButton2ActionPerformed(evt);
        }
    });

    javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
    jPanel1.setLayout(jPanel1Layout);
    jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup())
        .addContainerGap()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayou
t.Alignment.LEADING)
        .addComponent(jPanel2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addGroup(jPanel1Layout.createSequentialGroup()
        .addComponent(jButton1)

```

```

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)
                .addComponent(jButton2,
javax.swing.GroupLayout.DEFAULT_SIZE, 222, Short.MAX_VALUE)))
                .addContainerGap())
        );
        jPanell1Layout.setVerticalGroup(

jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.LEADING)
                .addGroup(jPanell1Layout.createSequentialGroup())
                .addComponent(jPanel2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)

.addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout
t.Alignment.BASELINE)
                .addComponent(jButton1)
                .addComponent(jButton2))

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
                .addGroup(layout.createSequentialGroup())
                .addContainerGap()
                .addComponent(jPanell1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
                .addGroup(layout.createSequentialGroup())
                .addContainerGap()
                .addComponent(jPanell1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

```

```

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
    );

    pack();
} // </editor-fold>

private void
jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    this.setVisible(false);
}

private void
jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    pf = new projectForm();
    pf.setVisible(true);
    this.setVisible(false);
}

public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {

        public void run() {
            new mainForm().setVisible(true);
        }

    });
}
// Variables declaration - do not modify
private javax.swing.JButton jButton1;
private javax.swing.JButton jButton2;
private javax.swing.JComboBox jComboBox1;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JPanel jPanel1;
private javax.swing.JPanel jPanel2;
private javax.swing.JTextField jTextField1;
// End of variables declaration
}
package spprdl;

public class projectForm extends javax.swing.JFrame {

    mainForm mf;
    processForm pf;

    public projectForm() {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();

```

```

jPanel2 = new javax.swing.JPanel();
jLabel1 = new javax.swing.JLabel();
jLabel2 = new javax.swing.JLabel();
jTextField1 = new javax.swing.JTextField();
jLabel3 = new javax.swing.JLabel();
jLabel4 = new javax.swing.JLabel();
jComboBox1 = new javax.swing.JComboBox();
jComboBox2 = new javax.swing.JComboBox();
jLabel5 = new javax.swing.JLabel();
jComboBox3 = new javax.swing.JComboBox();
jButton1 = new javax.swing.JButton();
jButton2 = new javax.swing.JButton();
jButton3 = new javax.swing.JButton();
jButton4 = new javax.swing.JButton();
jPanel3 = new javax.swing.JPanel();
jScrollPane1 = new javax.swing.JScrollPane();
jTextArea1 = new javax.swing.JTextArea();
jLabel6 = new javax.swing.JLabel();
jCheckBox1 = new javax.swing.JCheckBox();
jComboBox4 = new javax.swing.JComboBox();

```

```

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE
);

```

```

jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(jav
ax.swing.BorderFactory.createLineBorder(new java.awt.Color(0, 0,
0))), " Діагностика ");

```

```

jPanel2.setBorder(javax.swing.BorderFactory.createLineBorder(new
java.awt.Color(0, 0, 0)));

```

```

jLabel1.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N

```

```

jLabel1.setText("Номер картки:");

```

```

jLabel2.setText("ПІБ пацієнта:");

```

```

jLabel3.setText("Результати досліджень:");

```

```

jLabel4.setText("Скарги:");

```

```

jLabel5.setText("Діагноз:");

```

```

javax.swing.GroupLayout jPanel2Layout = new
javax.swing.GroupLayout(jPanel2);
jPanel2.setLayout(jPanel2Layout);
jPanel2Layout.setHorizontalGroup(

```

```

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.LEADING)
    .addGroup(jPanel2Layout.createSequentialGroup()
        .addContainerGap(

```

```

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(jPanel2Layout.createSequentialGroup()
            .addComponent(jLabel5)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jComboBox3, 0, 192,
Short.MAX_VALUE))
            .addComponent(jLabel11)

.addGroup(jPanel2Layout.createSequentialGroup()
            .addComponent(jLabel12)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jTextField1,
javax.swing.GroupLayout.DEFAULT_SIZE, 237, Short.MAX_VALUE))

.addGroup(jPanel2Layout.createSequentialGroup()
            .addComponent(jLabel13)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jComboBox2, 0, 239,
Short.MAX_VALUE))

.addGroup(jPanel2Layout.createSequentialGroup()
            .addComponent(jLabel14)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jComboBox1, 0, 221,
Short.MAX_VALUE)))
        .addContainerGap())
    );
jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel2Layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(jLabel11)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
            .addComponent(jLabel12)
            .addComponent(jTextField1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

```



```

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout
t.Alignment.BASELINE)
        .addComponent(jLabel13)
        .addComponent(jComboBox2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout
t.Alignment.BASELINE)
        .addComponent(jLabel14)
        .addComponent(jComboBox1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout
t.Alignment.BASELINE)
        .addComponent(jLabel15)
        .addComponent(jComboBox3,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
    );

    jButton1.setText("<");
    jButton1.addActionListener(new
java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent
evt) {
            jButton1ActionPerformed(evt);
        }
    });

    jButton2.setText("Дослідження");
    jButton2.addActionListener(new
java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent
evt) {
            jButton2ActionPerformed(evt);
        }
    });

    jButton3.setText("X");

```

```

        jButton3.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jButton3ActionPerformed(evt);
            }
        });

        jButton4.setText("Підібрати лікування");
        jButton4.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent
evt) {
                jButton4ActionPerformed(evt);
            }
        });

jPanel3.setBorder(javax.swing.BorderFactory.createLineBorder(new
java.awt.Color(0, 0, 0)));

        jTextArea1.setColumns(20);
        jTextArea1.setRows(5);
        jScrollPane1.setViewportView(jTextArea1);

        jLabel6.setFont(new java.awt.Font("Tahoma", 1, 11)); //
NOI18N
        jLabel6.setText("Альтернативи:");

        javax.swing.GroupLayout jPanel3Layout = new
javax.swing.GroupLayout(jPanel3);
        jPanel3.setLayout(jPanel3Layout);
        jPanel3Layout.setHorizontalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.LEADING)
            .addGroup(jPanel3Layout.createSequentialGroup()
                .addGroup(jPanel3Layout.createParallelGroup(
                    javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel6)
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.DEFAULT_SIZE, 153, Short.MAX_VALUE))
                .addGap(10, 10, 10)
                .addComponent(jCheckBox1)
                .addGap(10, 10, 10)
                .addComponent(jComboBox4,
javax.swing.GroupLayout.PREFERRED_SIZE, 97,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(10, 10, 10)
            )
        );
        jPanel3Layout.setVerticalGroup(

```

```

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.LEADING)
    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel3Layout.createSequentialGroup()
    .addContainerGap()
    .addComponent(jLabel6)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout
t.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel3Layout.createSequentialGroup()

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout
t.Alignment.LEADING)
    .addComponent(jCheckBox1)
    .addComponent(jComboBox4,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
    .addGap(91, 91, 91))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel3Layout.createSequentialGroup()
    .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 102,
javax.swing.GroupLayout.PREFERRED_SIZE)
    .addContainerGap()))
);

javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
jPanel1.setLayout(jPanel1Layout);
jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.LEADING)
    .addGroup(jPanel1Layout.createSequentialGroup()
    .addContainerGap()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout
t.Alignment.TRAILING, false)
    .addComponent(jPanel3,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
    .addComponent(jPanel2,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addGroup(javax.swing.GroupLayout.Alignment.LEADING,
jPanel1Layout.createSequentialGroup()

```

```

        .addComponent(jButton3)

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

        .addComponent(jButton1)

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

        .addComponent(jButton2,
        javax.swing.GroupLayout.DEFAULT_SIZE,
        javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        .addComponent(jButton4,
        javax.swing.GroupLayout.Alignment.LEADING,
        javax.swing.GroupLayout.DEFAULT_SIZE,
        javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))

        .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
        Short.MAX_VALUE))
        );
        jPanell1Layout.setVerticalGroup(

jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanell1Layout.createSequentialGroup()
        .addComponent(jPanel2,
        javax.swing.GroupLayout.PREFERRED_SIZE,
        javax.swing.GroupLayout.DEFAULT_SIZE,
        javax.swing.GroupLayout.PREFERRED_SIZE)

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

        .addComponent(jButton4)

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

        .addComponent(jPanel3,
        javax.swing.GroupLayout.DEFAULT_SIZE,
        javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

        .addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
        .addComponent(jButton3)
        .addComponent(jButton1)
        .addComponent(jButton2))
        .addContainerGap())
        );

        javax.swing.GroupLayout layout = new
        javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

```

```

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(layout.createSequentialGroup()
        .addContainerGap()
        .addComponent(jPanell1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
    );
layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(layout.createSequentialGroup()
        .addContainerGap()
        .addComponent(jPanell1,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        .addContainerGap())
    );

    pack();
} // </editor-fold>

private void
jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    this.setVisible(false);
}

private void
jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    mf = new mainForm();
    this.setVisible(false);
    mf.setVisible(true);
}

private void
jButton4ActionPerformed(java.awt.event.ActionEvent evt) {

}

private void
jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    pf = new processForm();
    this.setVisible(false);
    pf.setVisible(true);
}

public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {

        public void run() {
            new projectForm().setVisible(true);

```

```

        }
    });
}
// Variables declaration - do not modify
private javax.swing.JButton jButton1;
private javax.swing.JButton jButton2;
private javax.swing.JButton jButton3;
private javax.swing.JButton jButton4;
private javax.swing.JCheckBox jCheckBox1;
private javax.swing.JComboBox jComboBox1;
private javax.swing.JComboBox jComboBox2;
private javax.swing.JComboBox jComboBox3;
private javax.swing.JComboBox jComboBox4;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JLabel jLabel4;
private javax.swing.JLabel jLabel5;
private javax.swing.JLabel jLabel6;
private javax.swing.JPanel jPanel1;
private javax.swing.JPanel jPanel2;
private javax.swing.JPanel jPanel3;
private javax.swing.JScrollPane jScrollPane1;
private javax.swing.JTextArea jTextArea1;
private javax.swing.JTextField jTextField1;
// End of variables declaration
}
package spprd1;

public class processForm extends javax.swing.JFrame {

    public processForm() {
        initComponents();
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated
Code">
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        jPanel2 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jScrollPane1 = new javax.swing.JScrollPane();
        jTextArea1 = new javax.swing.JTextArea();
        jButton1 = new javax.swing.JButton();
        jButton2 = new javax.swing.JButton();
        jCheckBox1 = new javax.swing.JCheckBox();
        jComboBox1 = new javax.swing.JComboBox();
        jPanel3 = new javax.swing.JPanel();
        jLabel2 = new javax.swing.JLabel();
        jScrollPane2 = new javax.swing.JScrollPane();
        jTextArea2 = new javax.swing.JTextArea();
        jLabel3 = new javax.swing.JLabel();
        jScrollPane3 = new javax.swing.JScrollPane();
        jTextArea3 = new javax.swing.JTextArea();
        jButton3 = new javax.swing.JButton();

```

```

jCheckBox2 = new javax.swing.JCheckBox();
jComboBox2 = new javax.swing.JComboBox();
jButton4 = new javax.swing.JButton();
jButton5 = new javax.swing.JButton();
jButton6 = new javax.swing.JButton();
jButton7 = new javax.swing.JButton();
jButton8 = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE
);

jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(jav
ax.swing.BorderFactory.createLineBorder(new java.awt.Color(0, 0,
0)), " Лікування "));

jPanel2.setBorder(javax.swing.BorderFactory.createLineBorder(new
java.awt.Color(0, 0, 0)));

jLabel1.setText("Знайденні шаблони:");

jTextArea1.setColumns(20);
jTextArea1.setRows(5);
jScrollPane1.setViewportView(jTextArea1);

jButton1.setText("Затвердити лікування");

jButton2.setText("Корегувати шаблон");

javax.swing.GroupLayout jPanel2Layout = new
javax.swing.GroupLayout(jPanel2);
jPanel2.setLayout(jPanel2Layout);
jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.LEADING)
    .addGroup(jPanel2Layout.createSequentialGroup()
        .addGroup(jPanel2Layout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.TRAILING, false)
            .addComponent(jScrollPane1,
javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(jLabel1,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)
        .addGroup(jPanel2Layout.createParallelGroup(javax.swing.Grou
pLayout.Alignment.LEADING)
            .addComponent(jButton2,
javax.swing.GroupLayout.DEFAULT_SIZE, 193, Short.MAX_VALUE))
    )
    .addContainerGap()
)
)

```

```

.addGroup(jPanel2Layout.createSequentialGroup()
            .addComponent(jCheckBox1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)
            .addComponent(jComboBox1, 0, 170,
Short.MAX_VALUE))
        .addComponent(jButton1,
javax.swing.GroupLayout.DEFAULT_SIZE, 193, Short.MAX_VALUE))
        .addContainerGap())
    );
jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.LEADING)
        .addGroup(jPanel2Layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(jLabel1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayou
t.Alignment.LEADING)

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel2Layout.createSequentialGroup())

.addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayou
t.Alignment.TRAILING)
            .addComponent(jComboBox1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addComponent(jCheckBox1))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D, 40, Short.MAX_VALUE)
            .addComponent(jButton1)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)
            .addComponent(jButton2))
        .addComponent(jScrollPane1,
javax.swing.GroupLayout.DEFAULT_SIZE, 113, Short.MAX_VALUE))
        .addContainerGap())
    );

jPanel3.setBorder(javax.swing.BorderFactory.createLineBorder(new
java.awt.Color(0, 0, 0)));

jLabel2.setText("Згенеровані альтернативи:");

jTextArea2.setColumns(20);
jTextArea2.setRows(5);

```



```

jScrollPane2.setViewportViewView(jTextArea2);

jLabel3.setText("Порядок досліджень:");

jTextArea3.setColumns(20);
jTextArea3.setRows(5);
jScrollPane3.setViewportViewView(jTextArea3);

jButton3.setText("Корегувати");

jButton4.setText("Затвердити діагноз");

jButton5.setText("Зберегти у якості шаблону");

javax.swing.GroupLayout jPanel3Layout = new
javax.swing.GroupLayout(jPanel3);
jPanel3.setLayout(jPanel3Layout);
jPanel3Layout.setHorizontalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jPanel3Layout.createSequentialGroup()
        .addGap(
            .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                .addComponent(jButton5,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, 317, Short.MAX_VALUE)
                .addComponent(jButton4,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, 317, Short.MAX_VALUE)

            .addGroup(jPanel3Layout.createSequentialGroup()

            .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(jScrollPane2,
javax.swing.GroupLayout.DEFAULT_SIZE, 140, Short.MAX_VALUE)
                .addComponent(jLabel2,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))

            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

            .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(jScrollPane3,
javax.swing.GroupLayout.DEFAULT_SIZE, 171, Short.MAX_VALUE)
                .addComponent(jButton3,
javax.swing.GroupLayout.DEFAULT_SIZE, 171, Short.MAX_VALUE)
                .addComponent(jLabel3,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))

```

```

.addGroup(jPanel3Layout.createSequentialGroup()
            .addComponent(jCheckBox2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)
            .addComponent(jComboBox2, 0, 294,
Short.MAX_VALUE)))
        .addContainerGap())
    );
    jPanel3Layout.setVerticalGroup(

jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.LEADING)
        .addGroup(jPanel3Layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout
.Alignment.LEADING, false)

.addGroup(jPanel3Layout.createSequentialGroup()
            .addComponent(jLabel3)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)
            .addComponent(jScrollPane3)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)
            .addComponent(jButton3)
            .addComponent(jScrollPane2,
javax.swing.GroupLayout.PREFERRED_SIZE, 147,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)

.addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout
.Alignment.TRAILING)
        .addComponent(jCheckBox2)
        .addComponent(jComboBox2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)
        .addComponent(jButton4)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)
        .addComponent(jButton5)

```

```

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
    );

    jButton6.setText("<");

    jButton7.setText("X");

    jButton8.setText("Роздрукувати документацію");

    javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
    jPanel1.setLayout(jPanel1Layout);
    jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignmen
t.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()
            .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                        .addComponent(jButton6)
                        .addGroup(jPanel1Layout.createSequentialGroup()
                            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                            .addComponent(jButton8,
javax.swing.GroupLayout.DEFAULT_SIZE, 247, Short.MAX_VALUE)
                            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                            .addComponent(jButton7)))
                        .addContainerGap())
                    .addGap(10, 10, 10))
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addComponent(jPanel2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(10, 10, 10))
            )
        )
    );
    jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()
            .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                        .addComponent(jPanel2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGroup(jPanel1Layout.createSequentialGroup()
                            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                            .addComponent(jButton8)
                            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                            .addComponent(jButton7)))
                    .addGap(10, 10, 10))
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addComponent(jButton6)
                    .addGap(10, 10, 10))
            )
        )
    );

```

```

        .addComponent(jPanel3,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)

.addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout
t.Alignment.BASELINE)
        .addComponent(jButton6)
        .addComponent(jButton7)
        .addComponent(jButton8))

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
    );

    javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
    getContentPane().setLayout(layout);
    layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
        .addGroup(layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(jPanell1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
    layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
        .addGroup(layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(jPanell1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

    pack();
} // </editor-fold>

public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {

        public void run() {

```

```

        new processForm().setVisible(true);
    }
});
}
// Variables declaration - do not modify
private javax.swing.JButton jButton1;
private javax.swing.JButton jButton2;
private javax.swing.JButton jButton3;
private javax.swing.JButton jButton4;
private javax.swing.JButton jButton5;
private javax.swing.JButton jButton6;
private javax.swing.JButton jButton7;
private javax.swing.JButton jButton8;
private javax.swing.JCheckBox jCheckBox1;
private javax.swing.JCheckBox jCheckBox2;
private javax.swing.JComboBox jComboBox1;
private javax.swing.JComboBox jComboBox2;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JPanel jPanel1;
private javax.swing.JPanel jPanel2;
private javax.swing.JPanel jPanel3;
private javax.swing.JScrollPane jScrollPane1;
private javax.swing.JScrollPane jScrollPane2;
private javax.swing.JScrollPane jScrollPane3;
private javax.swing.JTextArea jTextArea1;
private javax.swing.JTextArea jTextArea2;
private javax.swing.JTextArea jTextArea3;
// End of variables declaration
}

```

ДОДАТОК В – Технічний опис

1 Загальні відомості

Назва: «СППР в медичній діагностиці».

Програмний продукт призначений для підвищення продуктивності праці лікарів та якості процесу лікування.

Для функціонування системи необхідні наступні програмні засоби:

- Операційна система Microsoft Windows 10 встановлена на робочих станціях та сервері;
- Наявність програмного забезпечення Microsoft Office 2013 та вище.
- Установлене на клієнтських місцях і на сервері ПЗ системи підтримки прийняття рішень.

Основним засобом розробки програмного забезпечення системи є програмне середовище NetBeans. Програмне забезпечення виконане мовою програмування Java.

2 Функціональне призначення

Система являє собою інструмент постановки діагнозу.

Створювана система реалізує наступні функції:

- Додавати інформацію про симптоми;
- Проводити пошук;
- Корегувати знайдений шаблон;
- Додавати інформацію про обстеження;
- Додавати інформацію про пацієнта;
- Затвердити діагноз;
- Друкувати документацію по проекту.

З урахуванням даних особливостей використання, основними функціональними характеристиками системи є:

- ведення загальної інформаційної бази з розподіленим доступом;
- авторизований доступ до даних на основі прав доступу;
- негайне відображення змін, внесених з будь-якого робочого місця;
- формування необхідних вихідних документів.

3 Опис логічної структури

При розробці програмного забезпечення системи використані методи об'єктно-орієнтованого проектування і розробки. Основною структурною одиницею ПЗ є клас, діаграма класів системи приведена в проектній документації. Основними частинами програмного забезпечення є клієнтські додатки та серверна частина. На клієнтські додатки покладена функція передачі даних на сервер, а безпосередню обробку інформації в базі даних забезпечує серверна частина. Клієнтська частина реалізує також графічний інтерфейс користувача з використанням стандартних віконних засобів операційної системи.

4 Використовувані технічні засоби

Для функціонування програмного забезпечення системи використовуються:

- Комп'ютери середньої комплектації на ступного типу: CPU Pentium E2200 3000 MHz; RAM 2048MB; HDD 320GB; VIDEO GeForce 7300 GT 256 DDR2; MONITOR 21”;
- Мережеве обладнання типу: LAN Ethernet Card Surecom, 10/100Mbps, RJ-45; Ethernet Switch 16-port 10/100Mbps Base TX;
- Сервер з характеристиками типу: CPU: 16 (32 core); RAM: 128 Gb; HDD: 500 Gb; Network Card: 2 (2 Gbit); Fiber Channel: 4
- Інше обладнання: принтери;

— Ефективного функціонування комп'ютерної мережі виробничих приміщень для зв'язку з сервером необхідне кабель – 100Base-FX. В середині комп'ютерних мереж офісу та приміщень цеху використовується тип кабелю 100Base-TX.

5 Вхідна інформація

Вхідною інформацією є інформація про деталь. Вхідні дані:

- скарги хворого;
- анамнез хвороби;
- анамнез життя;
- дані об'єктивного методу обстеження пацієнтів: загальний огляд пацієнта (загальний стан, положення в ліжку, свідомість, вираз обличчя, конституція, колір шкіри, стан волос та нігтів), пальпація (шкіри, підшкірної клітковини, лімфатичних вузлів), перкусія, аускультация;
- дані лабораторного методу дослідження: загальний аналіз, біохімічне та бактеріологічне дослідження. Особливості забору матеріалу в залежності від мети дослідження, підготовка пацієнта;
- дані інструментального методу дослідження: рентгенологічні, ендоскопічні, ультразвукові, радіоізотопні, комп'ютерні, у тому числі методи функціональної діагностики (ЕКГ, спірографія).

6 Вихідна інформація

Вихідною інформацією є згенерований діагноз.

ДОДАТОК Г – Програма та методика випробувань

Об'єкт випробувань.

Об'єктом випробувань є система підтримки прийняття рішень для в медичній діагностиці.

Мета випробувань.

Будь-який програмний продукт може мати в собі невиявлені помилки, оскільки дуже важко протестувати або передбачити реакцію чи поведінку його на будь-яких можливих комбінаціях вхідних даних, котрі вводить користувач. В загальному випадку під помилкою розуміють неправильність, похибку або навмисне спотворення об'єкта чи процесу. При цьому ми розуміємо чи правильний стан об'єкта чи процесу відомий. Попередньо, перед створенням завантажувального файлу, компілятор виправляє деякі помилки, але передбачити можливі значення різних змінних він не в змозі.

Цілі випробувань: перевірка придатності системи до використання по призначенню, її відповідність заданим вимогам і документації; перевірка працездатності і правильності роботи системи.

Вимоги до системи.

Вимоги до функціональних характеристик розробленої системи поставлені у технічному завданні (додаток А). Відповідно до цих вимог система забезпечує наступні функції:

- Додавати інформацію про симптоми;
- Проводити пошук;
- Корегувати знайдений шаблон;
- Додавати інформацію про обстеження;
- Додавати інформацію про пацієнта;
- Затвердити діагноз;
- Друкувати документацію по проекту.

Вимоги до програмної документації.

У комплект програмної документації повинні входити наступні документи:

- Загальний опис системи;
- Керівництво користувача;
- Технічне завдання;
- Програма і методика випробувань;
- Текст програми.

Засоби та порядок випробувань.

Вимоги до складу і параметрів технічних і програмних засобів вказані в технічному завданні (додаток А). Для правильної установки і настройки програмного забезпечення системи необхідно скористатися технічним описом системи (додаток Б) та інструкцією користувача (додаток Д). Сервера повинні бути об'єднані в комп'ютерну мережу з декількома клієнтами. Як мережне середовище можна використовувати кабелі типу «вита пара» категорії 3, 5 і вище. Мережне обладнання повинно підтримувати швидкість не менше 10 Мбіт/сек.

Випробування системи будуть проводитися протягом двох тижнів. Замовник і/або його представники самостійно проводять тестування усіх функцій системи в присутності розробника. При наявності відповідної можливості, у процесі тестування можуть додатково брати участь програмісти, тестувальники і системотехніки, які не брали участь у розробці системи.

Перед початком (не пізніше 3 днів) випробувань уточнюється кінцевий склад групи, що буде проводити випробування. Кожний член групи отримує необхідну документацію до системи (технічне завдання, загальний опис системи, керівництво користувача, інструкція з формування і ведення бази даних, керівництво адміністратора з установки програмного забезпечення на серверах, програма і методика випробувань).

Порядок проведення випробувань:

- Виконуються контрольні тести в довільному порядку. Результати заносяться у відповідний журнал.
- Робиться аналіз отриманих результатів і встановлюється відповідність програми заданим вимогам і системним документам.

При виявленні помилок у роботі системи складається їхній перелік і обговорюється термін їхнього виправлення розробником. Після цього замовник проводить повторне тестування в повному обсязі (можливо використання нових чи додаткових тестів).

Методика випробувань.

Процес тестування являє собою експлуатацію системи в контрольованих умовах і вивчення отриманих результатів. При цьому перевіряється робота системи з нормальними і помилковими даними і подіями. Тестування розподілених систем сильно ускладнено великою кількістю взаємозв'язків. Кожний елемент системи повинний тестуватися не тільки на коректність роботи, але і на сумісність один з одним у різних конфігураціях.

Найкраще для тестування розподілених систем застосовувати метод «знизу-нагору». Спочатку розробник повинен перевірити кожен компонент (елемент системи) окремо; потім – протестувати взаємодію компонента з іншими компонентами системи на автономному комп'ютері; і тільки потім – перевірити роботу всієї системи в розподіленому середовищі. Усі ці види тестування виконуються розробником системи без участі замовника. Розробник не зобов'язаний надавати тести і протоколи результатів тестування кому-небудь.

При прийманні системи проводяться випробування системи в розподіленому середовищі. Замовник і/або його представники самостійно проводять тестування усіх функцій системи методом «чорного ящика» у присутності розробника.

Програма випробувань

7 Тести складаються замовником і/або його представниками. При цьому перевіряється функціонування системи з нормальними і помилковими даними і подіями.

Наприклад:

- Авторизація користувача з правильними даними;
- Авторизація користувача з неправильними даними.

Під неправильними чи помилковими даними в контексті даної системи потрібно розуміти:

- Логін і/або пароль користувача, яких немає у файлах авторизації.
- Ведення даних неіснуючої моделі деталі.

Програмне забезпечення розроблено таким чином, що користувач не може ввести які-небудь інші неправильні дані в систему через наданий йому інтерфейс. Таким чином, будь-які дані, що не є неправильними, потрібно вважати правильними.

Для визначення переліку вхідних даних для кожного тесту можна скористатися формами для введення відповідних даних. У формах зазначені необхідні поля.

Вихідні дані чи результат при правильних даних повинні відповідати виконанню функції, для тестування якої створений тест; при неправильних даних — відмовлення в доступі до системи чи повідомлення про помилку.