ЗМІСТ

ВСТУП

Об’єктом дослідження дипломної роботи є телемедицина та телемедичні системи.

Телемидицина є відносно новим напрямком медичної практики, який робить можливим надання медичної допомоги на відстані і заснований на передачі спеціалізованої медичної інформації за допомогою сучасних інформаційно – телекомунікаційних технологій. Телемедицина дозволяє вирішити багато складних проблем, але найголовніше – вона дозволяє приблизити жителів віддалених регіонів країн до висококваліфікованих вітчизняних та закордонних медичних спеціалістів. Необхідність розвитку телемедичних систем на сьогоднішній день є очевидною для всіх провідних країн світу.

Термін «телемедицина», що вперше ввів ThomasBird в 1970 році, об’єднує безліч інформаційних і телекомунікаційних методів, що використовуються в сфері охорони здоров’я, а також їх різноманітні клінічні додаткки. Існує декілька десятків визначень телемедицини, які відрізняються в залежності від ступеню деталізації та різних аспектів, а також в залежності від вмісту технологій і напрямків що до неї входять.

Одне з визначень дав керівник телемедичного проекту в Портлендському дослідницькому центрі Н.Браун. Він визначив телемедицину як використання телекомунікацій для представлення медичної інформації та послуг, від обговорення клінічного випадку двома лікарями по телефону, до проведення інтерактивної відео-консультації між медичними центрами, що можуть знаходитись в різних точках світу з використанням супутникових технологій.

В 1997 році Всесвітня організація охорони здоров’я ввела більш широке поняття – медична телематика, що означає діяльність, послуги і системи, пов’язані з наданням медичної допомоги на відстані за допомогою інформаційно – телекомунікаційних технологій, направлених на розвиток системи охорони здоров’я, здійснення нагляду за епідеміями і надання медичної допомоги, а також навчання і проведення наукових медичних досліджень.

Найбільш вдалим визначенням на сьогоднішній день є таке: «Телемедицина – напрямок медицини, заснований на використанні сучасних комп'ютерних і телекомунікаційних технологій для адресного обміну медичною інформацією між спеціалістами з метою підвищення якості і доступності діагностики і лікування пацієнтів».

Телемедична система – сукупність програмних і апаратних засобів та комплексів, які реалізовують потенціал сучасних інформаційних технологій в сфері охорони здоров'я , а також відповідне фінансове та правове забезпечення.

На сьогоднішній день в багатьох країнах і міжнародних організаціях розробляються телемедичні проекти. ВООЗ розробляє проекти створення глобальної мережі телекомунікацій в медицині. Мається на увазі електронний обмін науковими документами і інформацією, її прискорений пошук, проведення відеоконференцій, нарад, консиліумів і електронного голосування. Уже сьогодні відомо більше ніж 250 телемедичних проектів, які за своїм характером поділяються на клінічні, навчальні, інформаційні і аналітичні. За географічним розподілом проекти поділяються на: місцеві (в межах одного закладу, їх частка 27%), регіональні – 40%, загальнонаціональні – 16%, міжнародні – 17%.

Також варто зазначити, що телемедичні системи мають значні перспективи розвитку в Україні, оскільки ведення електронних карт пацієнтів в медичних центрах та поліклініках значно полегшить роботу лікарів та медичного персоналу, а методи моніторингу пацієнтів допоможуть запобігти багатьом нещасним випадкам.

В рамках МОЗ України та представництвом ООН в Україні закладено основи національної телемедичної мережі, що передбачає оснащення спеціальним медичним обладнанням декілька телемедичних центрів , та їх об'єднання у єдину телемедичну мережу за допомогою високошвидкісних каналів передачі даних.

Предметом дослідження є використання методу аналізу ієрархій в задачах проектування та модернізації телемедичних систем.

Метод аналізу ієрархій був розроблений на початку 80-х років математиком Томасом Сааті, та широкого застосування набув лише за останнє десятиліття у

зв’язку з розвитком комп’ютерних інформаційних технологій. Він широко використовується у всьому світі для прийняття рішень в різноманітних ситуаціях у різних галузях, зокрема в бізнесі, промисловості, економіці, освіті, менеджменті, а від недавнього часу і в сфері охорони здоров’я.

Перевагою цього методу є те, що рішення можна використовувати для будь якої галузі, створюючи потрібні критерії відбору людиною-експертом. Значеннями ваг цих критеріїв будуть суб’єктивні погляди експерта чи групи експертів.

Задачею дипломної роботи є застосування методу аналізу ієрархії в області телемедичних систем. В даному випадку ця задача буде вирішена шляхом реалізації програмного продукту, який зможе допомогти в задачах проектування та модернізації телемедичних систем. В дипломній роботі буде розроблена ієрархія характеристик та підхарактеристик для обґрунтування вибору методів та засобів створення телемедичних систем, аналіз яких давав би об'єктивну оцінку щодо вибору тих чи інших інформаційних технологій та переваги застосування однієї альтернативи над іншими серед множини можливих.

Наукова новизна полягає у тому, що даний метод ще не був застосований в області телепедицини. Його впровадження в цю область допоможе приймати правильні рішення щодо вибору технологій проектування та модернізації телемедичних систем. Програмний продукт, що працює за алгоритмом МАІ допоможе заощадити час експертам при вирішенні даних питань.

РОЗДІЛ 1
ОСНОВНА ЧАСТИНА

1.1. Аналіз технічного завдання

Метою дипломної роботи є розробка програмного забезпечення для застосування методу аналізу ієрархії в задачах проектування та модернізації телемедичних систем. Для досягнення поставленої мети потрібно розв’язати комплекс таких взаємопов’язаних завдань, як:

1. Обґрунтувати застосування методу аналізу ієрархій в телемедичних системах та знайти його практичне застосування.

2. Створити якісну модель у вигляді ієрархії, яка б відображала основні критерії телемедичних систем, для забезпечення подальшого їх порівняння.

3. Розробити алгоритм, за яким можна проводити порівняння двох телемедичних систем, в залежності від ваг кожного з критеріїв, що задаються людиною – експертом, опираючись на особисті вподобання та вимоги до телемедичної системи.

4. Обрати програмну платформу та середовище розробки програмного продукту для подальшої реалізації у вигляді програмного коду, який буде містити математичні операції, необхідні для визначення кращої альтернативи. Як альтернативи будуть виступати телемедичні системи, які порівнюються.

5. Написання програмного продукту на основі отриманих даних, який би виконував поставлену задачу.

6. Потрібно провести економічні розрахунки, які будуть свідчити про економічну доцільність розробки.

Результатом виконання роботи, буде програма, яка порівнювати між собою телемедичні системи за їхніми критеріями, яким задані відповідні ваги. Робота програми повинна бути оптимізованою для забезпечення високої продуктивності та швидких обчислень. Результати обробки даних повинні бути подані у зручному для користувача інтерфейсі.

Необхідно провести огляд телемедичних систем, їх типи, моделі, приклади та методи опрацювання даних, що до них надходять, з метою вибору та обґрунтування важливих критеріїв і характеристик телемедичних систем, які будуть в подальшому проходити порівняння.

1.2. Огляд телемедичних систем

1.2.1. Загальна інформація про телемедичні системи

Телемедицина є одним з перспективних напрямків медицини, який заснований на використанні інноваційних комп’ютерних і телекомунікаційних технологій для обміну інформацією між спеціалістами, пацієнтами та медичними центрами з метою підвищення якості, доступності діагностики і лікування. Також під телемедициною розуміють сукупність засобів забезпечення на великих відстанях медичною інформацією будь якого об’єкту який забезпеченій комп’ютерним обладнанням. На сьогоднішній день для здійснення цієї задачі вже є розроблені всі технічні засоби. Дальній зв'язок (телезв’язок) забезпечений сучасними супутниковими системами радіозв’язку, наземними каналами оптоволоконного і провідного зв’язку на досить високому технічному рівні. Збір, обробка, зберігання даних і відображення інформації також забезпечується сучасними комп’ютерними системами. В якості системи яка передає та приймає дані найчастіше використовується міжнародна інформаційна мережа Internet.

Іншими словами, телемедицина – сукупність вбудованих в медичні інформаційні системи нових засобів обробки даних, що об’єднуються в цілісні технологічні системи, які в свою чергу, забезпечують створення, передавання, зберігання та відображення інформаційного продукту (даних, знань) з найменшими витратами з метою проведення потрібних лікувальних та діагностичних заходів, а також навчання для всіх, що його потребує, в потрібному місці в потрібний час[1].

Телемедичні системи є ефективним інструментом для вирішення таких медичних завдань:

- надання допомоги лікарям, які працюють у віддалених стаціонарних та тимчасових медичних пунктах;

- полегшення поширення методичних і управлінських документів в структурі охорони здоров'я;

- передавання знань і досвіду фахівців провідних медичних центрів лікарям, що проходять навчання, віддалене проведення кваліфікаційних іспитів і сертифікацій лікарів;

- надання необхідних інструментів для вирішення завдань діагностики і лікування хворих, підвищення кваліфікації лікарів, збирання і поширення управлінської інформації;

- об'єднання всіх регіональних об'єктів ОЗ в єдиний телемедичний простір.

Телемедична система являє собою сукупність засобів та комплексів, які реалізують потенціал сучасних інформаційних та телекомунікаційних технологій в охороні здоров'я, а також відповідне фінансове та правове забезпечення.

Телемедичні системи використовуються в усіх галузях сучасної медицини:

- клінічна медицина – діагностика, визначення тактики лікування, трактування результатів обстежень, допомога в прийнятті рішень і т.д.;

- організація – створення інформаційних мереж, координація дій різних установ, ведення реєстрів тощо;

- навчальний процес – дистанційне навчання, використання матеріалів телеконсультацій у педагогічному процесі і. т. д.

У складі системи можна виділити чотири типи елементів, взаємодія яких і утворює телемедичну систему:

- консультаційний центр – медичні організації, що мають в штаті висококваліфікованих лікарів у різних напрямках медицини та інформаційні ресурси, медичні діагностичні пристрої, бази даних для проведення дистанційних консультацій, консиліумів та лікувально-діагностичних процедур;

- технічні засоби доступу в телекомунікаційні мережі;

- каналоутворювальне середовище – набір апаратних, програмних засобів, носіїв інформації та технологічних рішень (протоколи та стандарти), що забезпечують передавання різнорідної інформації в територіально розподіленому середовищі;

- диспетчерський пункт – датчики та інші перетворювачі медичної інформації в цифрові електричні сигнали для передавання по каналах зв'язку.

При необхідності в структурі телемедичних систем формуються тимчасові структури – наприклад, комплекс віддалених медичних підрозділів у місцях бойових дій чи техногенних катастроф. Такі станції розгортаються та підключаються до телемедичних систем з метою залучення груп досвідчених спеціалістів провідних центрів для рішення оперативних проблем, які виникають.

Ще одним необхідним елементом телемедичної системи є служба мобільної телемедичної допомоги, для якої віддалені станції розгортаються на основі транспортних засобів – автомобілів, авіазасобів, засобах водного та залізничного транспорту.

Актуальність розвитку телемедичних систем обґрунтована активним розвитком інформаційних технологій, що допоможе значно полегшити обмін медичною інформацією на великих відстанях, проводити моніторинг хворих, навчання лікарів, проведення операцій та об’єднати всі регіональні телемедичні центри в єдиний інформаційний простір.

Також актуальність впровадження телемедицини в Україні диктується специфікою системи охорони здоров’я, яскраво вираженою різницею в рівні матеріального забезпечення і підготовки спеціалістів в центральних і віддалених регіонах, специфікою системи фінансування. Одним з основних аспектів застосування телемедицини є значне скорочення бюджетних ресурсів всіх рівнів на надання діагностичної, консультаційної і лікувальної допомоги пацієнтам, особливо у віддалених територіях.

Телемедичні системи можуть покращити якість медичної допомоги за рахунок розширення спектру послуг, в тому числі і в найбільш розвинутих галузях

медицини, таких як радіологія, діаліз, кардіохірургія, психіатрія, дерматологія як для груп населення так і для окремих осіб.

1.2.2. Задачі телемедицини

До задач телемедицини можна віднести:

- профілактичне обслуговування населення;
- обслуговування віддалених суб'єктів;
- підвищення рівня обслуговування;
- зниження вартості медичних послуг (за рахунок використання комунікаційних технологій зменшуються витрати на відрядження, витрати на забезпечення служб "швидкої допомоги", скорочується термін перебування пацієнта в стаціонарі і т. д.).

В арсеналі телемедицини є безліч технічних засобів, в тому числі і телефон, радіо, модеми, спеціалізовані сканери, спеціалізовані системи відображення відеографічної інформації, а також пристрої суміщення комп'ютерних та спеціалізованих медичних приладів.

Медичну допомогу можна надавати як в реальному часі (наприклад, за допомогою інтерактивного відео), так і з затримкою (передавання текстових чи графічних даних, фотографій, коротких відеокліпів).

Для використання в телемедичних системах оптимально підходять спеціалізовані медичні прилади, які мають візуальний чи акустичний зворотний зв'язок з лікарем, а також вбудовану системну підтримку. Обладнання та канали забезпечують передавання різнорідної інформації – алфавітно-цифрової та графічної, відео- та аудіопотоків, а також цифрових та аналогових сигналів, що знімаються з датчиків, і передаються на органи управління діагностичною та лікувальною апаратурою.

Слід відзначити, що єдиного методу, який підходить для вирішення всіх задач телемедицини, не існує; технічні характеристики кожної системи визначаються виходячи з потреб користувачів.

Телемедичні програмно-апаратні комплекси призначені для проведення телеконсультацій і телеконсиліумів, дистанційної діагностики, моніторингу складних медичних маніпуляцій з використанням відеозв'язку в режимі реального часу. Все обладнання розроблене з урахуванням підтримки медичних стандартів із зберігання і передавання різного роду медичної інформації і даних в різних форматах. Телемедичне обладнання, що адаптоване для роботи в операційних приміщеннях і стійке до електромагнітного або рентгенівського випромінювання, джерелом якого є інше медичне устаткування, застосовується для здобуття необхідних даних для встановлення діагнозу пацієнта. Операційні програмно-апаратні комплекси ефективно використовуються для проведення операцій, що дає можливість повноцінного обміну всією медичною інформацією з провідними фахівцями інститутів і спеціалізованих клінік, як результат – більш кваліфіковане проведення операції. Операційні телемедичні комплекси знайшли активне застосування як в спеціалізованих медичних центрах, де відбуваються щодня різні за рівнем складності операції і виникає необхідність консультування у момент її проведення, так і операційних клінічних установ районного і обласного значення.

Мобільний телемедичний комплекс призначений для надання оперативної дистанційної консультативно-діагностичної медичної допомоги. Мобільний телемедичний комплекс складається з комп'ютерного, телекомунікаційного обладнання, за допомогою якого можна провести первинну діагностику стану пацієнтів і отримати дистанційну консультативну допомогу. До мобільних телемедичних комплексів відносяться телемедичні рішення на базі літаків, вертольотів, де окрім консультацій з телемедичного устаткування можна надавати повноцінну медичну допомогу різного ступеня складності.

Мобільні телемедичні системи – це компактні мобільні телемедичні прилади, за допомогою яких виконують повноцінне медичне консультування і діагностику. Як правило, все телемедичне обладнання такого класу розроблене з урахуванням використання його в умовах підвищеної небезпеки і має міцний водонепроникний, протиударний і вогнестійкий корпус, а програмне забезпечення системи є стійким.

Такі системи отримали позитивну оцінку фахівців військової медицини і служби порятунку різних країн Європи і світу.

Розвиток телемедицини особливо важливий в кардіології, оскільки небезпека серцево-судинних захворювань часто полягає в гострому несподіваному початку й атиповій клінічній картині, що не дозволяє самим пацієнтам, а часто і дільничним лікарям поліклінік швидко й правильно оцінити ситуацію, що у свою чергу може призвести до смерті пацієнта.

Найважливішим методом правильної діагностики серцево-судинних захворювань є реєстрація ЕКГ. Тому широкого розвитку набула система дистанційної цілодобової невідкладної консультативної кардіологічної допомоги з можливістю безпосереднього передавання ЕКГ по телефонних лініях. Це дозволяє в 3 – 9 разів скоротити час із моменту появи перших симптомів захворювання до надання кваліфікованої медичної допомоги в повному обсязі, що в багатьох випадках допомагає зберегти життя й здоров'я пацієнтів.

Приймальні станції впроваджені на базі звичайних сучасних персональних ЕОМ і дозволяють відтворювати, обробляти й зберігати як самі ЕКГ, так і необхідну медичну інформацію.

Надання спеціалізованої консультативно-діагностичної допомоги кардіологічним хворим забезпечується декількома шляхами:

- приймання та консультація електрокардіограм по телефону;

- телефонні консультації лікарів лінійних бригад з діагностики та тактики лікування, транспортування хворих, вибору стаціонару для госпіталізації;

- виїзд спеціалістів центру до хворого додому або в громадські місця для діагностики та консультативної роботи з подальшим наданням медичної допомоги за показаннями та госпіталізацією до медичних установ.

Впровадження в роботу телемедицини шляхом організації діагностично-консультаційних центрів допоможе забезпечити:

- екстрену кваліфіковану діагностику для населення, яке звертається за медичною допомогою;

- можливість проведення диспансеризації (ранньої діагностики) хворих;

- архівацію обстежень та забезпечення автоматичного збереження електронних файлів в електронних базах даних з можливістю порівняння даних в динаміці;

- відпрацювання системи раннього виявлення і подальшого лікування хворих з захворюваннями, що загрожують життю, у тому числі в пацієнтів із хронічними хворобами.

Як додаткові результати впровадження сучасних телемедичних технологій в установи охорони здоров'я можна також відзначити:

- значне підвищення якості надання медичної допомоги пацієнтам, які знаходяться на будь-якій відстані від провідних клінічних центрів;

- раціональне використання праці висококваліфікованих лікарів;

- скорочення термінів тимчасової непрацездатності населення, зменшення кількості випадків виходу на інвалідність за рахунок своєчасного надання екстреної медичної допомоги;

- якісно новий рівень надання медичної допомоги у фельдшерських і лікарських пунктах охорони здоров’я;

- різке зниження вартості отримання висококваліфікованої медичної допомоги за рахунок виключення міжміських переїздів.

Рішення для телемедицини поділяються на такі групи:

1. Телемедичні консультації.
2. Відкладені телеконсультації.
3. Консультації в режимі реального часу.
4. Теленавчання.
5. Трансляція хірургічних операцій.
6. Мобільні телемедичні комплекси.
7. Системи дистанційного біомоніторингу.
8. Домашня телемедицина.

Телемедичні консультації здійснюються шляхом передачі медичної інформації по телекомунікаційних каналах зв'язку. Консультації можуть проводитися як в відкладеному режимі, так і в режимі реального часу.

Відкладені телеконсультації - це найбільш дешевий і простий спосіб організації консультації на відстані передачі медичної інформації по електронній пошті. Він мало підходить для екстрених випадків, однак не вимагає значних витрат і досить ефективний при належному організаційному забезпеченні процесу.

Консультації в режимі реального часу більш вимогливі до технічного оснащення, їх проводять з використанням широкосмугових каналів зв'язку та відеоапаратури. Розрізняють планові, екстрені відеоконсультації і відеоконсиліуми. У всіх цих випадках забезпечується безпосереднє спілкування між консультантом і лікарем. Найчастіше такі консультації проводяться з участю хворого. При цьому сеанс відеозв'язку може проходити як між двома абонентами, так і між декількома абонентами в так званому багатоточковому режимі, тобто найбільш складні випадки можуть обговорюватися консиліумом лікарів з різних медичних центрів. Телемедичні системи дозволяють організувати діалог з лікарем-експертом (відеоконференцію) на будь-якій відстані і передати практично всю необхідну для кваліфікованого висновку медичну інформацію (виписки з історії хвороби, рентгенограми, комп'ютерні томограми, знімки УЗД і так далі.).

Теленавчання: проведення лекцій, відеосемінари, конференцій з використанням телекомунікаційного обладнання. Під час таких лекцій викладач може мати інтерактивний контакт з аудиторією. В результаті використання таких технологій у лікаря з'явилася реальна можливість безперервної професійної освіти без відриву від місця роботи. Лекції, як і відеоконсультації можуть проходити в багатоточковому режимі, таким чином, лекція може бути прочитана відразу для слухачів з декількох регіонів.

Трансляція хірургічних операцій. Застосування мережевих відеокамер дозволяє організувати трансляцію хірургічної операції. Дана технологія може використовуватися також в режимі «наставника», коли більш досвідчений лікар дистанційно контролює дії менш досвідченого колеги в режимі реального часу.

Мобільні телемедичні комплекси призначені для роботи на місцях аварій. Малогабаритні мобільні діагностичні комплекси можна використовувати у відсутності телемедичних кабінетів і центрів, безпосередньо там, де виникла необхідність. Цими коштами доцільно оснащувати і машини швидкої допомоги, і сімейних лікарів, районні та сільські лікарні, бригади медицини катастроф та санітарної авіації, медичні формування МНС та підрозділів МОЗ. Сучасний мобільний телемедичний комплекс об'єднує в собі потужний комп'ютер, що легко сполучаються з різноманітним медичним обладнанням, засоби ближнього і дальнього бездротового зв'язку, засоби відеоконференції та засоби IP-передачі даних.

Системи дистанційного біомоніторингу, або так звані телемедичні системи динамічного спостереження, застосовуються для спостереження за пацієнтами, що страждають хронічними захворюваннями, а також на промислових об'єктах для контролю стану здоров'я працівників (наприклад, операторів на атомних електростанціях). Багатообіцяючим напрямком розвитку таких систем є інтеграція датчиків в одяг, різні аксесуари, мобільні телефони. Наприклад, жилет з набором датчиків, що реєструють ЕКГ, артеріальний тиск і ряд інших параметрів, або мобільний телефон з можливістю реєстрації ЕКГ і відправки її засобами GPRS в медичний центр, а також з можливістю визначення координат людини у разі загрози життю.

Домашня телемедицина являє собою дистанційне надання медичної допомоги пацієнту, що проходить курс лікування в домашніх умовах. Спеціальне телемедичне обладнання здійснює збір і передачу медичних даних пацієнта з його будинку у віддалений телемедичний центр для подальшої обробки фахівцями. Це важливо, наприклад, для хворих із серцевою недостатністю, які потребують регулярних і частих обстеженнях. Комплекси, що включають датчики, які вимірюють температуру тіла, тиск крові, парціальний тиск кисню, ЕКГ і функції дихання, з'єднані з настільним монітором, який, в свою чергу, автоматично відправляє дані в телемедичний центр.

1.2.3. Приклади телемедичних систем

Телемедична система КМІС (Комплексна медична інформаційна система). Ця система створена з ціллю автоматизації закладів охорони здоров'я для обміну медичною інформацією між спеціалістами для підвищення якості діагностики і лікування пацієнтів[2].

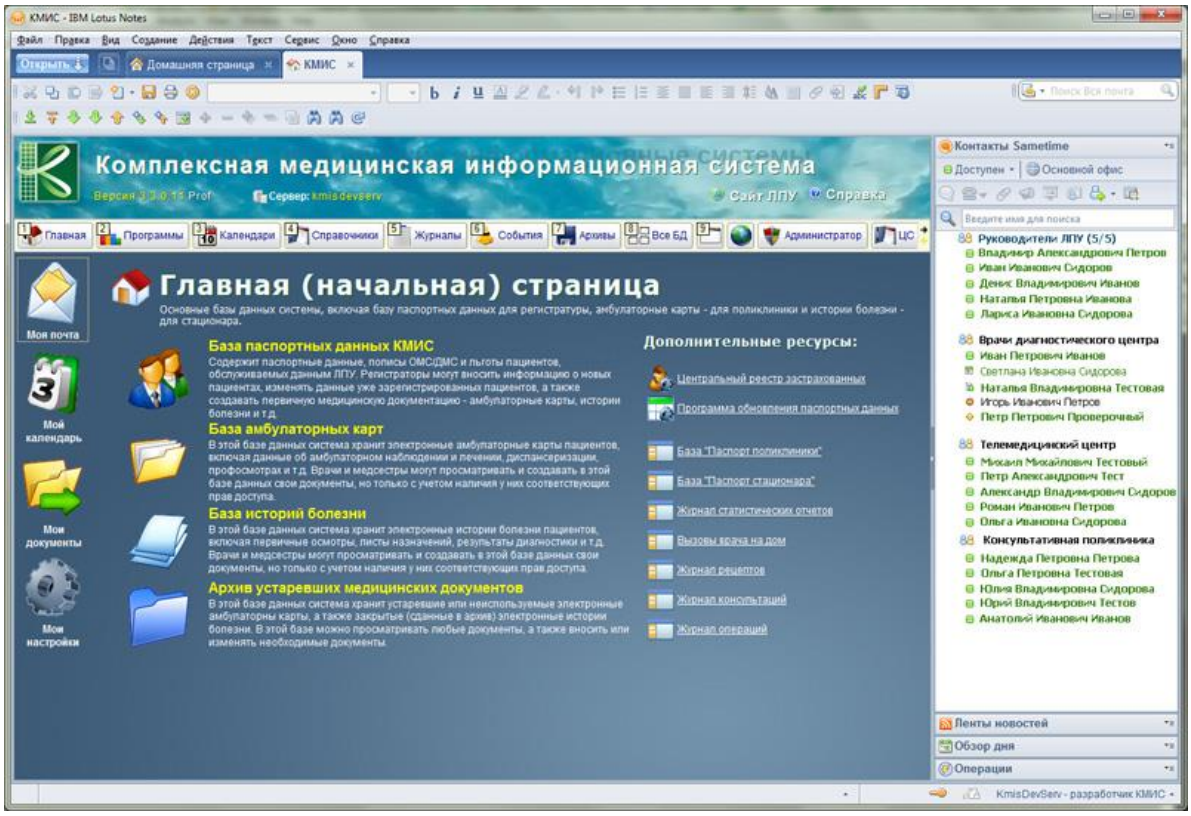

Рис.1.1. Початкова сторінка КМІС

Телемедична система КМІС призначена для вирішення наступних задач.

1. Проведення телемедичних консультацій – можливість обмінюватися медичною інформацією телекомунікаційними каналами зв'язку в режимі «реального часу». Важливість цієї задачі останнім часом тільки зростає.

2. Відкладені телеконсультації – можливість обмінюватися медичними записами через електронну пошту чи систему інформаційного обміну КМІС. Застосовується в тих випадках, коли канали зв'язку не дозволяють використовувати відео в режимі реального часу.

3. Теленавчання – можливість проведення лекцій, відеосемінарів, конференцій і т.д. з віддаленою демонстрацією електронної медичної карти і відеоконференцзв'язком між лікарями і аудиторією. Дозволяє надати лікарям можливість безперервного професійного навчання без відривання від місця роботи. Лекції, як і відеоконсультації можуть проходити в багатоточковому режимі, таким чином, лекція може бути прочитана відразу для багатьох слухачів з різних регіонів.

4. Спільне обговорення медичної карти пацієнта – можливість підключити декількох віддалено розташованих спеціалістів і обговорити з ними результати обслідування або лікування пацієнтів, в тому числі – зі спільним доступом до електронної карти пацієнта.

Телемедична система КМІС базується на платформі IBM Sametime з сімейства продуктів IBM Lotus. Для розвертання телемедичної системи КМІС потрібний видалений сервер, на якому встановлюється спеціальний блок сервісів і служб, що відповідають за роботу всього модуля. В комплект поставки системи входить вся документація, що полегшує встановлення і налаштування платформи IBM Sametime. При цьому серверна частина IBM Sametime тісно інтегрована з основним сервером КМІС IBM Lotus Domino – в тому числі вона використовує єдиний каталог користувачів та тісно інтегрована з почтовою системою і клієнтським програмним забезпеченням IBM Lotus Notes.

Для роботи користувачів і впровадження телемедичних технологій підтримується зразу декілька можливостей:

1. Клієнт, що вбудований в користувацьке програмне забезпечення КМІС для роботи з телемедичним сервером (спільна робота з телемедичною системою прямо з інтерфейсу КМІС). Клієнтське ПЗ телемедичної системи функціонує під управлінням Microsoft Windows, Linux, Mac OS X.

2. Web-клієнт, що дозволяє підключитися до телемедичних консультацій віддаленим користувачам, що не мають на своєму ПК клієнтське ПЗ КМІС (віддалене підключення до телемедичного серверу з будь якої точки світу). Підтримуються всі найбільш популярні браузери.

3. Мобільні клієнти для роботи з телемедичною системою КМІС зі смартфонів і планшетних ПК. Підтримуються пристрої під управлінням Apple iOS (iPhone, iPad), BlackBerry, Windows Mobile, Android.

Телемедична система КМІС володіє наступними функціями:

1. Розвиток можливості відеоконференцзв'язку. Підтримуються свої власні або інтегровані засоби для передавання голосових даних (VoIP), відео високої якості, передача відео по протоколу point-to-point.

2. Сповіщення про присутність: встановивши телемедичну систему КМІС можна бачити фізичну присутність лікарів на своїх місцях, бачити статус (зайнятий, вільний і т.д.), бачити статус доступності телефону – для того щоб взнати хто може допомогти прямо зараз і який засіб зв'язку краще всього використовувати, підтримка автоматичного визначення місцезнаходження – якщо, для прикладу, співробітник рухається між підрозділами, можна взнати в якому місці він зараз знаходиться. При виході користувача з системи або зайнятості, коли він не працює з системою, спрацьовує автоматична зміна статусу.

3. Система миттєвого обміну повідомленнями. Для тих користувачів, хто звик або зацікавлений в миттєвому обміні повідомленнями на зразок того, як це реалізовано в програмі Skype – телемедчна система КМІС надає таку можливість, проте вже повністю інтегровану в клієнтське ПЗ системи. Користувач може вибрати потрібного співробітника і відправити миттєве повідомлення – яке буде виведене в окремому вікні на екрані. Підтримується автоматичне зберігання історії чату, значки емоцій, вбудований редактор RichText, автоматична перевірка на орфографію, багатокористувацький чат і інші зручні функції.

4. Система обміну файлами - підтримується можливість спілкуватися і обмінюватися файлами за допомогою захищено системи обміну повідомленнями корпоративного рівня. Підтримується функція створення знімків екрану, в тому числі і можливість додавання коментарів до створених знімків: вона дозволяє, для прикладу, зробити знімок томографічного зображення з робочої станції лікаря-діагноста, обвести потрібне місце на зображенні (наприклад патологічне

утворення), додати свій коментар і відправити це лікуючому лікарю або потрібному спеціалісту для додаткового інформування або телеконсультації.

Рис.1.2. Приклад передачі знімку з додаванням коментарів

5. Проведення інтерактивних зібрань – дозволяє організувати захищене приватне обговорення, до якого можна підключати віддалених користувачів і в режимі конференції або чату обмінюватися думками, обговореннями і т. д. Підтримується віддалений доступ до робочого столу для обговорення результатів діагностичного обслідування або спільного обговорення електронної медичної карти.

6. IP-телефонія – ця функція дозволяє перетворити ПК медичного співробітника в інтелектуальний телефон з можливістю інтеграції з існуючою інфраструктурою телефонії.

7. Відправка оголошень – спеціальна функція, що дозволяє відправити групове оголошення користувачам, щоб повідомити або нагадати важливу

інформацію – наприклад запросити на нараду з головним лікарем або оголосити про якусь екстрену ситуацію.

8. Інтеграція – дозволяє використовувати телемедичний сервіс КМІС в тісній інтеграції з іншими інформаційними системами. Наприклад, підтримується інтеграція з IBM WebSphere, Microsoft Office, Microsoft Outlook, Microsoft SharePoint, Microsoft ActiveDirectory та ін. В наявності є відкриті API-інтерфейси для підключення невідомих систем. Забезпечена підтримка інтеграції з Jabber, XMPP, в тому числі підтримка TLS при підключенні до спільнот XMPP (якщо конкретний сервер XMPP підтримує цю функцію), а також можливість додавання проксі сервера XMPP в конфігурацію кластера. За допомогою спеціального інтеграційного шлюзу користувачі можуть повідомляти про свій стан і обмінюватися повідомленнями з корисувачами систем AOL AIM, ICQ, Apple iChat, Yahoo Messenger, Google Talk, додаткові ліцензії на програмне забезпечення при цьому не потрібні.

Медична система «Медіалог» розроблена для вирішення комплексу лікувальних і управлінських задач, що стоять перед сучасними поліклініками, та являє собою повноцінний продукт – робочий інструмент для керівників, лікарів і всіх співробітників медичного закладу[3].

Дана система має наступні переваги.

1. Єдиний інформаційний простір. Він забезпечує оперативний доступ до інформації і допомагає організувати ефективну роботу медичного закладу. Завдяки використанню даної системи, медична, фінансова і інша інформація стає доступна для співробітників клініки в режимі реального часу. Для прикладу, лікар може ознайомитися з результатами лабораторних досліджень як тільки вони надійшли в Медіалог, прийняти рішення про подальше лікування і зробити відповідне направлення. Наявність єдиного інформаційного простору дозволяє економити час цінних спеціалістів, швидко приймати обґрунтовані рішення, краще контролювати процеси і прогнозувати роботу клініки в цілому.

2. Ведення електронної медичної карти. Процес надання медичної допомоги відображається в електронній медичній карті пацієнта, яка об’єднує в собі історію

хвороби і амбулаторну карту. З моменту реєстрації вся медична інформація заноситься в Медіалог співробітниками клініки. Електронна карта являє собою структуроване сховище для зберігання інформації. Кожний лікар працює з налаштованими для своєї спеціальності медичними протоколами. В системі передбачені зручні механізми вводу і перегляду даних, що підвищують швидкість роботи і прозорість представлення інформації.

3. Ефективне планування і контроль використання ресурсів. Система дозволяє полегшити задачу планування і оптимізації ресурсів клініки – часу спеціалістів, лабораторного і діагностичного обладнання, місць, палат та ін. Використовуючи електронний розклад, лікарі самостійно планують час прийому і деякі дослідження, що значно економить час співробітників медичного закладу і пацієнтів. Контроль використання ресурсів допомагає уникнути накладок в часі та раціонально організувати роботу спеціалістів і проведення досліджень.

4. Оперативна підготовка документів. Це одна з ключових переваг медичної карти – оперативність отримання медичної документації та статистичної звітності. Час лікарів значно економиться завдяки можливості роздрукувати будь які документи, довідки, направлення, результати обстежень та ін. Крім паперових документів медична інформаційна система передбачає можливість експорту медичної карти в електронному виді (в відкритому форматі HTML) з подальшим записом на флеш-носій або інший носій зручного формату. Це дозволяє лікарям інших медичних закладів отримати доступ до електронної карти пацієнта.

5. Інтегрована лабораторна інформаційна система. Медична інформаційна система Медіалог також автоматизує роботу лабораторії, що значно збільшує її продуктивність і підвищує пропускну здатність. Наявність в системі лабораторного модулю скорочує витрати медичного закладу на автоматизацію, надаючи можливість використання єдиної інформаційної системи, як в лікувальних, так і в діагностичних підрозділах без необхідності інтеграції різного роду програмних продуктів. Для швидкого підключення аналізаторів і інших приборів існує широка бібліотека драйверів.

6. Інтеграція з іншими рішеннями. Мова йде про інтеграцію з іншими інформаційними системами і медичним обладнанням. Наприклад інтеграція з професійними диктофонами (Philips), PACS – системами, IP-телефонією, Call-центром (CISCO), різним діагностичним обладнанням.

Медична інформаційна система Медіалог підтримує міжнародні медичні стандарти HL7 і DICOM.

В 1996 році Американським національним інститутом стандартів (ANSI) в США був затверджений національний стандарт обміну медичними даними в електронному вигляді – HL7.

HL7 (Health Level 7) – стандарт обміну, керування і інтеграції електронної медичної інформації. «Сьомий рівень» - аналогія з вищим комунікаційної моделі відкритих систем (OSI). Сьомий рівень підтримує виконання таких задач, як:

- структурування даних, що передаються;
- можливості проектування систем;
- досягнення узгодження передач;
- безпека;
- ідентифікація учасників;
- доступність.

Стандарт HL7 призначений для полегшення взаємодії комп'ютерних додатків в установах охорони здоров'я. Його основна мета полягає в такій стандартизації обміну даними між медичними комп'ютерними додатками, при якій виключається або зменшується необхідність розробки і реалізації певних програмних інтерфейсів, які зазвичай є необхідними за умови відсутності стандарту. Він легко сполучається та взаємодіє з іншими протоколами, що дозволяє використовувати його в приладах багатьох виробників світу [4].

Цей стандарт широко застосовується в різних установах охорони здоров'я. Єдина схема отриманих даних є цінною властивістю не тільки для клінічних, але й для статичних досліджень.

Загальна структура стандарту включає:

- рух пацієнтів (надходження, виписка, переведення);

- порядок прибуття;

- контроль фінансових питань;

- дані клінічних спостережень;

- інтерфейс для даних загального призначення;

- інформацію для керівного персоналу;

- призначення, операції і лікувальні процедури.

DICOM (Digital Imaging and Communication in Medicine) – окремий стандарт створення, зберігання, передачі і візуалізації медичних зображень і документів пацієнтів, що проходять обстеження.

DICOM опирається на ISO-стандарт OSI, підтримується основними виробниками медичного обладнання і медичного програмного забезпечення.

Стандарт DICOM, розроблений національною асоціацією виробників електронного обладнання (National Electrical Manufacturers Association), дозволяє створювати, зберігати, передавати і друкувати окремі кадри зображень, серії кадрів, інформацію про пацієнтів, дослідження, обладнання, закладах, медичному персоналі та ін.

Стандартом DICOM визначено два інформаційних рівні:

- файловий рівень – DICOM File – об’єктний файл з теговою організацією для представлення кадру зображення.

- мережевий (Комунікаційний) – DICOM Network Protocols (мережевий DICOM-протокол) – для передачі DICOM файлів і команд по мережах з підтримкою протоколу TCP/IP.

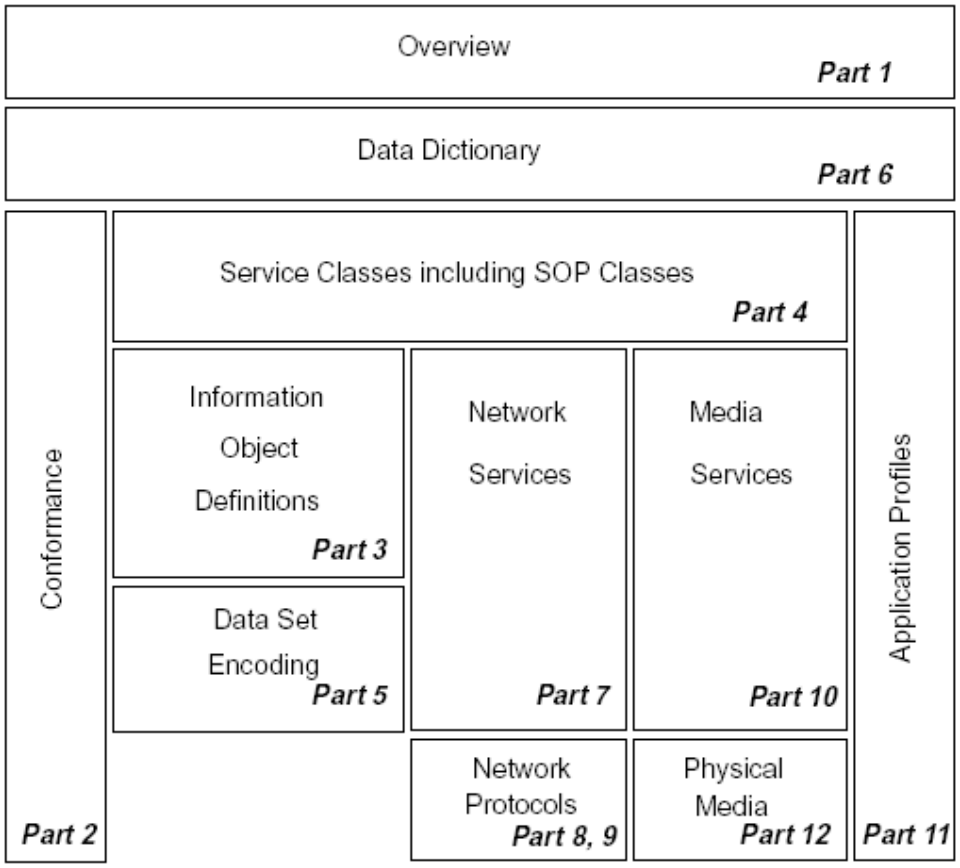

Рис.1.3. Структура стандарту DICOM

1.3. Метод аналізу ієрархій

Метода аналізу ієрархій (MAI) – математичний інструмент системного підходу до складних проблем прийняття рішень. MAI дозволяє особі, що приймає рішення, в інтерактивному режимі знайти такий варіант (альтернативу), який найкраще узгоджується з розумінням суті проблеми і потребами до її вирішення. Цей метод розроблений американським математиком Томасом Мааті, який написав про нього книги і розробив програмні продукти. MAI широко використовується на практиці і активно розвивається вченими всього світу. В його основу закладені математичні і психологічні аспекти. Цей метод дозволяє зрозумілим і раціональним чином структурувати складну проблему прийняття рішень в виді ієрархії, порівняти і виконати кількісну оцінку альтернативних варіантів рішення[5]. Метод аналізу ієрархій використовується для прийняття рішень в різноманітних ситуаціях – від керування на міждержавному рівні до рішення конкретних питань в бізнесі,

промисловості, охороні здоров'я і освіті. Для комп'ютерної підтримки МАІ існують програмні продукти, розроблені різними компаніями. Аналіз проблеми прийняття рішень в МАІ починається з побудови ієрархічної структури, яка включає ціль, критерії, альтернативи і інші фактори, що впливають на вибір. Ця структура відображає розуміння проблеми особою, що приймає рішення. Кожен елемент ієрархії може представляти різні аспекти задачі, що вирішується, при чому до уваги можуть бути прийняті як матеріальні так і не матеріальні фактори, вимірювані кількісні параметри і якісні характеристики, об'єктивні дані і суб'єктивні експертні оцінки. Іншими словами, аналіз ситуації вибору рішення в МАІ нагадує процедури і методи аргументації, які використовуються а інтуїтивному рівні. Наступним етапом аналізу є визначення пріоритетів, що представляють відносну важливість або перевагу елементів побудованої ієрархічної структури за допомогою попарних порівнянь. Безрозмірні пріоритети дозволяють обґрунтовано порівнювати різного роду фактори, що є основною особливістю МАІ. В заключному етапі аналізу виконується синтез пріоритетів ієрархії, в результаті якого визначаються пріоритети альтернативних рішень відносно головної цілі. Кращою вважається альтернатива з максимальний значенням пріоритету.

Метод аналізу ієрархії був створений для полегшення процесів рішень в різних сферах людської діяльності. Оскільки процеси прийняття рішень в різних сферах є дуже подібними, доцільно розробити універсальний метод підтримки прийняття рішень, що відповідає логічному мисленню людини [7].

Часто економічні, медичні, політичні, соціальні, адміністративні проблеми мають декілька варіантів вирішення. Часто, вибираючи одне рішення серед можливих, особа, що приймає рішення, керується лише інтуїтивним представленням проблеми. В результаті прийняте рішення має певну невизначеність, що відображається на якості рішення, що приймається.

З метою усунення цієї невизначеності, процес підготовки прийняття рішення на всіх етапах супроводжується кількісним вираженням таких категорій, як «перевага», «важливість», «бажаність» та ін.

Задачі прийняття рішень можна розглянути наступним чином.

Нехай є:

1. декілька однотипних альтернатив (об’єктів, дій та ін.);

2. основний критерій (основна ціль) порівняння альтернатив;

3. декілька груп однотипних факторів, що впливають на вибір альтернатив.

Потрібно кожній альтернативі поставити у відповідність пріоритет (значення) – отримати рейтинг альтернатив. При цьому чим більшу перевагу має альтернатива за вибраним критерієм, тим більший її пріоритет.

Задачі прийняття рішень зазвичай найбільш часто постають перед:

- працівниками адміністрацій;
- економістами;
- фінансистами;
- соціологами;
- політиками;
- консультантами;
- працівниками охорони здоров’я;
- воєнними офіцерами;
- психологами;
- працівниками соціальної сфери;
- науковими працівниками.

Далі наведені приклади задач, що можуть вирішуватися за допомогою MAI.

1. Рейтинг клієнтів.
2. Аналіз ризиків.
3. Розподіл ресурсів.
4. Планування від досягнутого та для майбутнього.
5. Комбіноване планування.
6. Вибір оптимальної стратегії.
7. Аналіз ефективності та вартості.
8. Прийняття кадрових рішень.
9. Вирішення конфліктів.

10. Пошук існуючих факторів.

11. Діагностика.

12. Побудова залежностей.

1.3.1. Принцип роботи алгоритму методу аналізу ієрархій

Метод аналізу ієрархій (MAI) був створений математиком Томасом Сааті і на даний момент широко використовується для вибору єдиного компромісного рішення з урахуванням різних критеріїв у різних сферах і набув широкого розповсюдження в останнє десятиріччя. Згідно з ним методом, вибір рішення здійснюється шляхом попарних порівнянь за допомогою відповідної шкали[6].

Метод аналізу ієрархій містить процедуру синтезу пріоритетів, що обчислюються на основі суб'єктивних критеріїв експертів. Кількість критеріїв може вимірюватись в десятках або навіть в сотнях. Математичне обчислення для задач невеликої розмірності можна виконувати вручну або за допомогою калькулятора, проте найбільш зручно використовувати програмне забезпечення (ПЗ) для вводу і обробки критеріїв. Найпростіший спосіб комп'ютерної підтримки – електронні таблиці, найбільш розвинене ПЗ передбачає застосування спеціальних пристроїв для вводу критеріїв учасниками процесу колективного вибору. Порядок застосування методу аналізу ієрархій:

1. Побудова якісної моделі проблеми в виді ієрархії, що включає в себе ціль, альтернативні варіанти досягнення цілі і критерії для оцінки якості альтернатив.

2. Визначення пріоритетів всіх елементів ієрархії з використанням методу попарних порівнянь.

3. Синтез глобальних пріоритетів альтернатив шляхом лінійного згортання пріоритетів елементів на ієрархії.

4. Перевірка суджень на узгодженість.

5. Прийняття рішення на основі отриманих результатів.

1.3.2. Математичний опис методу аналізу ієрархій

1. Отримуємо попарні порівняння (ω_i/ω_j), проведені експертами та проводимо апроксимацію кожного критерію.

$$a_i = \sqrt[n]{\frac{\omega_i}{\omega_1} \cdot \ldots \cdot \frac{\omega_i}{\omega_n}} \tag{1.1}$$

2. Обчислюємо нормалізовану наближену оцінку локального пріоритету окремого критерію (x_i) на критерії вищого рівня ієрархії.

$$x_i = \frac{a_i}{\sum_{j=1..n} a_j} \tag{1.2}$$

3. Для перевірки адекватності одержаних результатів знайдемо індекс узгодженості матриці (*CI*), максимальне власне значення ($\lambda_{max}>0$), індекс послідовних співвідношень (*CR*):

$$\lambda_{max} = \sum_{i=1..n} x_i \cdot s_i \tag{1.3}$$

$$CI = \frac{\lambda_{max} - n}{n - 1} \tag{1.4}$$

$$CR = \frac{CI}{RI} \tag{1.5}$$

де s_i – сума елементів стовбця з номером i матриці попарних порівнянь, n – кількість альтернатив, *RI* – випадковий індекс матриці. На основі одержаного значення індексу послідовних співвідношень робимо висновок про адекватність

експертних оцінок, даний показник не повинен перевищувати 10%, для систем, що не вимагають особливої точності – 20% .

4. Обчислюємо глобальний пріоритет альтернативи як суму добутків локальних пріоритетів компонентів на шляху ієрархії від компоненту останнього рівня до першого.

1.3.3. Переваги і недоліки методу аналізу ієрархій

В рамках методу аналізу ієрархій немає загальних правил для формування структури моделі прийняття рішення. Це є відображенням реальної ситуації прийняття рішення, оскільки завжди для одної і тої ж проблеми існує ціла множина можливих суджень. Метод дозволяє врахувати цю обставину за допомогою побудови додаткової моделі для узгодження різних думок, шляхом визначення їх пріоритетів. Таким чином, метод дозволяє враховувати «людський фактор» при підготовці прийняття рішення. Це одна з головних переваг цього методу перед іншими методами прийняття рішень.

Формування структури моделі прийняття рішення в методі аналізу ієрархії – достатньо трудомісткий процес. Проте в результаті можна отримати детальне представлення про те, як саме взаємодіють фактори, що впливають на пріоритети альтернативних рішень і самі рішення, а також про те, як саме формуються рейтинги можливих рішень і рейтинги, що відображають важливість факторів. Процедури розрахунків рейтингів в методі аналізу ієрархій є достатньо простими, що сильно відрізняє даний метод від інших методів прийняття рішень.

Збір даних для підтримки прийняття рішення здійснюються головним чином за допомогою процедури попарних порівнянь. Результати попарних порівнянь можуть дещо протирічити один одному. При цьому виникає необхідність повторного перегляду даних для мінімізації протирічь. В свою чергу процедури попарних порівнянь і процес повторного перегляду результатів порівнянь для мінімізації протирічь часто є досить трудомісткими процесами. Проте в результаті,

особа, що приймає рішення, отримує впевненість в тому, що дані є достатньо коректними.

В методі аналізу ієрархій, єдиним засобом для перевірки достовірності даних є коефіцієнт послідовності даних. Проте цей недолік обумовлений специфікою використання методу. Якщо збір даних був проведений за допомогою досвідчених експертів, і в даних немає значних протирічь, то якість таких даних можна признати задовільною.

Схема застосування методу зовсім не залежить від сфери діяльності, в якій приймається рішення. Тому метод є універсальним, його застосування дозволяє організувати систему підтримки прийняття рішень.

Робота, що стосується підготовки до прийняття рішення є занадто трудомісткою для однієї людини. Модель, створена за допомогою методу аналізу ієрархії, завжди має кластерну структуру. Застосування методу дозволяє розбити велику задачу на ряд невеликих самостійних задач. Завдяки цьому для підготовки прийняття рішення можна залучити експертів, що працюють незалежно один від одного над локальними задачами. Експерти можуть не знати нічого про характер рішення, що приймається. Завдяки цьому вдається зберегти таємну інформацію про рішення, що буде прийматися.

Метод аналізу ієрархії відображає роботу людської логіки і дає більш загальний підхід, ніж інші запропоновані методи. Він надає не тільки спосіб виявлення найбільш бажаного рішення, але і дозволяє кількісно виразити ступінь переваги, завдяки рейтингу. Це сприяє повному виявленню переваг особи, що приймає рішення. Крім того коефіцієнт послідовності даних допомагає визначити міру коректності введених даних.

1.4. Розробка алгоритму методу аналізу ієрархій

1.4.1. Побудова моделі в виді ієрархії

При проведенні планування та підготовки телемедичної системи, розробники повинні врахувати вимоги користувачів щодо її роботи та зробити акценти на певних характеристиках.

Зазвичай, в методі аналізу ієрархій, на першому рівні знаходиться кінцева ціль, в нашому випадку – вибір телемедичної системи.

В якості глобальних критеріїв (другий рівень), будуть розглядатися наступні аспекти телемедичної системи.

1. Моніторинг – функція телемедичної системи, яка полягає у спостереженні за станом об’єкту для визначення і передбачення погіршення стану або для збирання потрібної інформації. По суті моніторинг передбачає систематичний збір інформації, яка надалі буде проходити обробку відповідними технічними і програмними засобами для правильної інтерпритації. В плані моніторингу за станом здоров’я пацієнта, в якості даних що надходять, можуть бути кардіо-сигнали (кардіограма, частота сердечних скорочень), для виявлення порушень в роботі серцево–судинної системи та своєчасного реагування.

2. Передача даних – функція, що являє собою фізичний перехід даних (цифрового бітового потоку) в виді сигналів від точки до точки, або від точки до декількох точок засобами зв’язку по відповідних каналах зв’язку, для подальшого зберігання і обробки засобами обчислювальної техніки і програмними засобами. Прикладами каналів зв’язку можуть бути мідні кабелі, оптоволокно, засоби бездротового зв’язку. Вибір каналу зв’язку в телемедичних системах зазвичай залежить від масштабу системи. Якщо система існує тільки в межах одного телемедичного центру або поліклініки то достатньо буде провести кабелі витої пари, якщо система об’єднує декілька медичних закладів, які розташовані в межах одної географічної локації – необхідним буде використання оптоволоконних

кабелів. У випадку необхідності передачі даних на далекі відстані – буде використовуватися супутниковий зв'язок.

3. Діагностика – передбачає набір технічних та програмних засобів, наявних у телемедичній системі, які використовуються в процесі встановлення діагнозу, тобто висновку про природу і тип хвороби, стан пацієнта, у відповідній медичній термінології. Діагностика базується на використанні пристроїв, що слідкують за роботою. До таких пристроїв можна віднести:

- апарат магнітно-резонансної томографії (МРТ), який дозволяє використовувати метод дослідження внутрішніх органів і тканин з використанням фізичного явища ядерного магнітного резонансу;

- електрокардіографи – пристрої, що реєструють і досліджують електричні поля, що з’являються при роботі серця;

- електронні тонометри – пристрої для вимірювання артеріального тиску;

- установки для проведення ультразвукової діагностики;

- датчики, монітори та інше обладнання.

4. Збереження даних – передбачає збереження отриманих даних від пацієнтів, лікарів, інших телемедичних центрів на надійних носіях інформації, наприклад на серверах телемедичного центру. Також збереження даних передбачає ведення баз даних для обліку пацієнтів, що надходять у медичний центр, лікарів, технічного обладнання. До цієї ж категорії входить забезпечення захисту та конфіденційності цих даних та створення копій у разі необхідності для запобігання втрат важливих даних у разі технічних проблем.

5. Опрацювання даних – передбачає обробку даних, що надходять в основному від пацієнтів, за якими ведеться спостереження, обробку зовнішнього та внутрішнього трафіку телемедичного центру. Якістю опрацювання даних можна вважати кількість даних, що обробляються за одиницю часу. Для швидкого опрацювання даних необхідна потужна обчислювальна техніка, здатна встигати обробити весь трафік, що циркулює в межах телемедичної системи.

Існують різні вимоги до телемедичних систем, зокрема в розрахунок беруться наступні характеристики:

- Інформативність.
- Точність.
- Ефективність.
- Вартість.
- Функціональність.
- Багатозадачність.
- Передача даних.
- Зберігання даних.
- Захищеність даних.
- Доступність.
- Стабільність.
- Продуктивність.
- Надійність.
- Масштабованість.
- Інтегрованість.
- Мобільність.

В нашому випадку, на другому рівні ієрархії будуть розташовані глобальні критерії, такі як: Моніторинг(A1), Передача даних(A2), Діагностика(A3), Збереження даних(A4), Опрацювання даних(A5).

Серед наведених вище характеристик, для наступного рівня нашої ієрархії, візьмемо такі: Інформативність(B1), Продуктивність(B2), Вартість(B3), Функціональність(B4), Надійність(B5), Масштабованість(B6).

Інформативність – характеристика системи, що виражає придатність ознак для проведення діагностики стану пацієнта.

Продуктивність – величина, яка характеризує швидкість обробки одиниці інформації за одиницю часу.

Вартість – характеристика, що вказує на величину вкладених грошових коштів, необхідних для встановлення, розгортання, роботи та модернізації телемедичної системи.

Функціональність – характеристика, що описує можливості телемедичної системи.

Надійність – можливість системи зберігати параметри, необхідні для виконання функцій в заданих режимах і умовах застосування, захищеність системи, стійкість до збоїв.

Масштабованість – можливість змін та розвитку телемедичної системи без значних змін в самій її архітектурі.

При цьому кожен з критеріїв ієрархії другого рівня, буде зв’язаний з кожним критерієм ієрархії третього рівня. Це означає, що кожен критерії другого рівня буде мати підкритерії на третьому рівні, а глобальний пріоритет критерію третього рівня буде дорівнювати сумі добутків локальних пріоритетів третього рівня на відповідні їм локальні пріоритети матриць вищого (другого) рівня.

Відповідно четвертий рівень ієрархії складається з критеріїв, що є підкритеріями до третього рівня.

Інформативність: діагностичні ознаки, сигнал, метод.

Продуктивність: багатозадачність, обчислювальна потужність, точність.

Вартість: розгортання, обслуговування, модернізація.

Функціональність: технічна підтримка, навчання, представлення.

Надійність: неінвазивність, захищеність, відновлення.

Масштабованість: інтегрованість, здатність до збільшення, мобільність.

П’ятий рівень ієрархії є останнім. На ньому зазвичай розміщуються альтернативи, які проходять порівняння. В нашому випадку будуть порівнюватися дві альтернативи, я якості яких будуть виступати телемедичні системи за власним бажанням особи, що приймає рішення. В свою чергу, ми будемо порівнювати дві телемедичні системи: КМІС і СП.Арм. Це і будуть наші альтернативи.

Таким чином ми отримаємо наступну ієрархію:

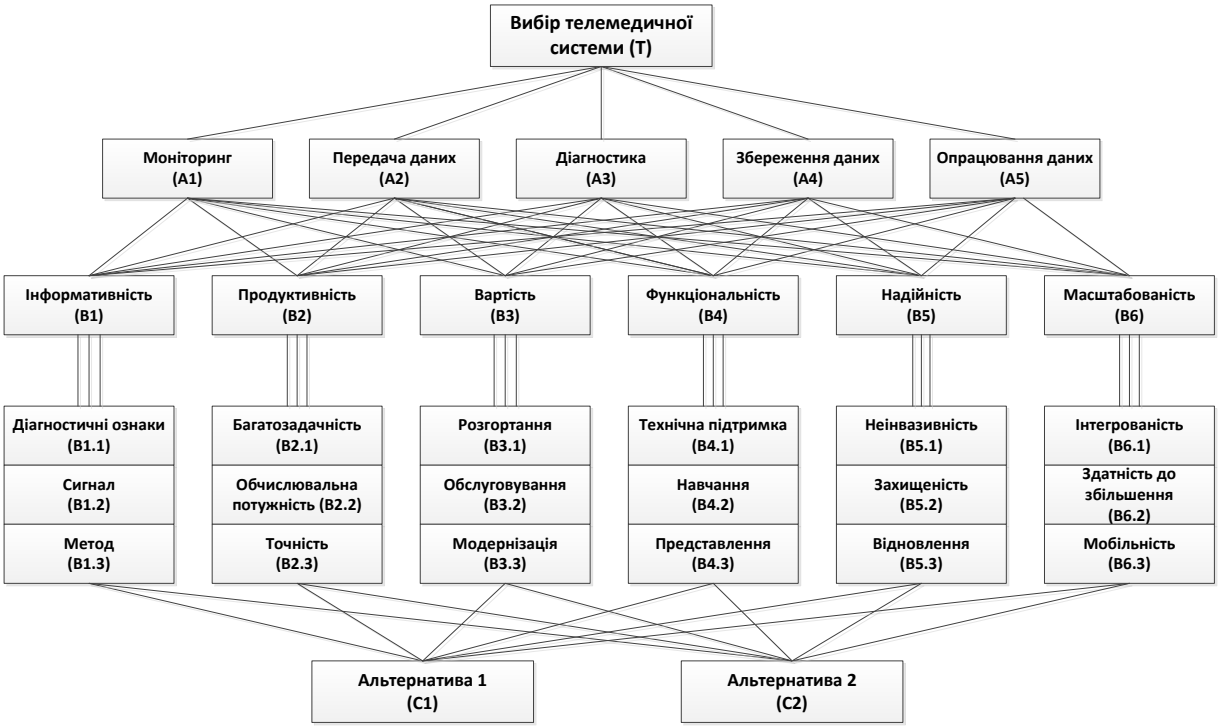

Рис.1.4 – Ієрархія вибору телемедичної системи

1.4.2. Визначення пріоритетів всіх елементів ієрархії

Для визначення пріоритетів всіх елементів в методі аналізу ієрархії використовується шкала експертних порівнянь, при оцінюванні переваги першого об’єкту над другим зі значеннями від 1 до 9. Зміст цих оцінок показано в табл 1.1.

Таблиця 1.1.

Ступені важливості для матриць попарних порівнянь

Ступінь важливості	Означення	Коментар
1	Однакова важливість	Обидва об'єкти вносять однаковий внесок у досягнення мети
3	Слабка значущість	Досвід і судження надають легку перевагу першому об'єкту перед другим
5	Суттєва значущість	Досвід і судження надають сильну перевагу першому об'єкту перед другим

Продовж.Табл.1.1.

7	Дуже сильна значущість	Перевага першого об'єкта над другим дуже сильна
9	Абсолютна значущість	Свідчення на користь переваги першого об'єкту у вищому степені переконлива
2,4,6,8	Проміжні значення між сусідніми значеннями шкали	Ситуації, коли потрібні компромісні рішення
Обернені величини наведених значень	Якщо для порівняння об'єкта А з об'єктом В отримане одне з наведених значень	

Далі будуємо матриці попарних порівнянь за кожним критерієм та обрахуємо числові характеристики цих матриць – вектор значень (який власне буде містити локальні пріоритети кожної матриці) та індекс узгодженості матриці для перевірки точності введених даних.

Побудову матриць будемо здійснювати, рухаючись від першого рівня ієрархії і до останнього (п’ятого). Відповідно перша матриця буде містити попарні порівняння критеріїв другого рівня, відносно головної мети (першого рівня).

В нашому випадку, телемедична система повинна володіти надійною системою моніторингу, для запобігання нещасних випадків з пацієнтами, тому критерій «моніторинг» буде мати перевагу над іншими критеріями в тій чи іншій степені. «Передача даних» також є важливим критерієм, оскільки у випадку її ненадійності телемедичний центр втратить зв'язок з пацієнтами та іншими центрами, що може мати негативні наслідки. «Опрацювання даних» в даному випадку буде мати невелику перевагу над «збереженням даних», оскільки телемедичному центру потрібна певна обчислювальна потужність для того щоб справлятися з потоком даних.

Таблиця 1.2.

Матриця попарних порівнянь для вибору альтернатив за критерієм «вибір телемедичної системи»

Вибір телемедичної системи	Моніторинг	Передача даних	Діагностика	Збереження даних	Опрацювання даних
Моніторинг	1	2	3	5	3
Передача даних	0,5	1	0,5	3	3
Діагностика	0,333	2	1	3	4
Збереження даних	0,2	0,333	0,333	1	0,5
Опрацювання даних	0,333	0,333	0,25	2	1

Обчислення проведено способом Томаса Сааті (див.[5], стор 24-25).

Ваги критеріїв будуть мати наступні значення:

- моніторинг – 0,402;
- передача даних – 0,192;
- діагностика – 0,247;
- збереження даних – 0,066;
- опрацювання даних – 0,091.

Знаходимо оцінку найбільшого власного значення

$$\lambda_{max} = 0{,}402 * (1 + 0{,}5 + 0{,}333 + 0{,}2 + 0{,}333) +$$
$$0{,}192 * (2 + 1 + 2 + 0{,}333 + 0{,}333) + 0{,}247 * (3 + 0{,}5 + 1 + 0{,}333 + 0{,}25) +$$
$$0{,}66 * (5 + 3 + 3 + 1 + 2) + \ 0{,}901 * (3 + 3 + 4 + 0{,}5 + 1) = 5{,}2833$$

Знаходимо індекс узгодженості

$$\mathrm{CI} = \frac{5{,}2833 - 5}{4} = 0{,}0708$$

Знаходимо індекс послідовності співвідношень

$$CR = \frac{0{,}0708}{1{,}12} = 0{,}0632$$

, де 1,12 – випадковий індекс, для n=5. Таблиця випадкових індексів виглядає наступним чином:

Таблиця 1.3.

Випадкові індекси

n	1	2	3	4	5	6	7	8	9	10
index	0,00	0,00	0,58	0,90	1,12	1,24	1,32	1,41	1,45	1,51

Далі проводимо попарні порівняння для наступного рівня ієрархії. В даному випадку будуть порівнюватись критерії третього рівня (інформативність, продуктивність, вартість, функціональність, надійність, масштабованість) відносно головної мети – критеріїв другого рівня. Оскільки між елементами ієрархії другого і третього рівня існує повний зв'язок, то всі кожен з критеріїв третього рівня буде розглядатися відносно кожного з критеріїв другого рівня.

Таблиця 1.4.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Моніторинг»

Моніторинг	Інф.	Прод.	Варт.	Функ.	Над.	Масш.
Інформативність	1	2	3	3	2	8
Продуктивність	0,5	1	3	2	1	5
Вартість	0,3333	0,3333	1	0,5	0,3333	3
Функціональність	0,3333	0,5	2	1	3	6
Надійність	0,5	1	3	0,3333	1	4
Масштабованість	0,125	0,2	0,333	0,16667	0,25	1

Таблиця 1.5.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Передача даних»

Передача даних	Інф.	Прод.	Варт.	Функ.	Над.	Масш.
Інформативність	1	3	5	3	0,3333	5
Продуктивність	0,3333	1	2	1	0,2	2
Вартість	0,2	0,5	1	3	0,25	1
Функціональність	0,3333	1	0,3333	1	0,5	3
Надійність	3	5	4	2	1	6
Масштабованість	0,2	0,5	1	0,3333	1,66667	1

Таблиця 1.6.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Діагностика»

Діагностика	Інф.	Прод.	Варт.	Функ.	Над.	Масш.
Інформативність	1	4	2	3	1	2
Продуктивність	0,25	1	1	0,5	0,2	2
Вартість	0,5	1	1	0,3333	0,2	0,3333
Функціональність	0,3333	2	3	1	0,3333	2
Надійність	1	5	5	3	1	7
Масштабованість	0,5	0,5	3	0,5	1,1428	1

Таблиця 1.7.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Збереження даних»

Збереження даних	Інф.	Прод.	Варт.	Функ.	Над.	Масш.
Інформативність	1	3	2	0,5	0,3333	1
Продуктивність	0,3333	1	1	0,3333	0,2	2
Вартість	0,5	1	1	0,5	0,2	0,5

Продовж.Табл.1.7.

Функціональність	2	3	2	1	0,2	1
Надійність	3	5	5	5	1	2
Масштабованість	1	0,5	2	1	0,5	1

Таблиця 1.8.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Опрацювання даних»

Опрацювання даних	Інф.	Прод.	Варт.	Функ.	Над.	Масш.
Інформативність	1	3	2	0,5	0,3333	3
Продуктивність	0,3333	1	1	0,3333	0,2	2
Вартість	0,5	1	1	0,5	0,2	0,5
Функціональність	2	3	2	1	1	5
Надійність	3	5	5	1	1	4
Масштабованість	0,3333	0,5	2	0,2	0,25	1

Зробивши розрахунки, способом, аналогічному тим, яким ми обчислювали ваги критеріїв попереднього рівня, отримаємо значення локальних пріоритетів для третього рівня ієрархії, відносно критеріїв другого рівня. Вони будуть мати наступні значення:

Для моніторингу: інформативність – 0,343; продуктивність – 0,210; вартість – 0,082; функціональність – 0,180; надійність – 0,150; масштабованість – 0,035.

Для передачі даних: інформативність – 0,343; продуктивність – 0,210; вартість – 0,082; функціональність – 0,180; надійність – 0,150; масштабованість – 0,035.

Для діагностики: інформативність – 0,343; продуктивність – 0,210; вартість – 0,082; функціональність – 0,180; надійність – 0,150; масштабованість – 0,035.

Для збереження даних: інформативність – 0,343; продуктивність – 0,210; вартість – 0,082; функціональність – 0,180; надійність – 0,150; масштабованість – 0,035.

Для опрацювання даних: інформативність – 0,343; продуктивність – 0,210; вартість – 0,082; функціональність – 0,180; надійність – 0,150; масштабованість – 0,035.

1.4.3. Синтез глобальних пріоритетів

Наступним кроком буде розрахунок глобальних пріоритетів критеріїв. Він передбачає побудову матриць попарних порівнянь для передостаннього рівня ієрархії. По суті це останній рівень ієрархії, в якому знаходяться критерії, тому що на п’ятому рівні в MAI розташовані альтернативи, серед яких нам потрібно зробити вибірку. Таким чином ми знайдемо глобальний пріоритет кожного кінцевого критерію для подальшого порівняння альтернатив.

На цьому рівні, кожен з критеріїв третього рівня (інформативність, продуктивність, вартість, функціональність, надійність, масштабованість) має по 3 підкритерії четвертого рівня. Будуємо для них матриці попарних порівнянь:

Таблиця 1.9.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Інформативність»

Інформативність	Діагн. ознаки	Сигнал	Метод
Діагн. ознаки	1	3	2
Сигнал	0,3333	1	0,5
Метод	0,5	2	1

Таблиця 1.10.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Продуктивність»

Продуктивність	Діагн. ознаки	Сигнал	Метод
Діагн. ознаки	1	3	0,3333
Сигнал	0,3333	1	0,2
Метод	3	5	1

Таблиця 1.11.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Вартість»

Вартість	Діагн. ознаки	Сигнал	Метод
Діагн. ознаки	1	2	5
Сигнал	0,5	1	1
Метод	0,2	1	1

Таблиця 1.12.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Функціональність»

Функціональність	Діагн. ознаки	Сигнал	Метод
Діагн. ознаки	1	0,5	0,2
Сигнал	2	1	1
Метод	5	1	1

Таблиця 1.13.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Надійність»

Надійність	Діагн. ознаки	Сигнал	Метод
Діагн. ознаки	1	5	4
Сигнал	0,2	1	2
Метод	0,25	0,5	1

Таблиця 1.14.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Масштабованість»

Масштабованість	Діагн. ознаки	Сигнал	Метод
Діагн. ознаки	1	2	3
Сигнал	0,5	1	1
Метод	0,3333	1	1

Провівши розрахунки, отримаємо значення глобальних пріоритетів кінцевих критеріїв, представлених у табл.1.5.

Таблиця 1.15.

Глобальні пріоритети

Інформативність (0,277)	Діагностичні ознаки	0,149
	Сигнал	0,045
	Метод	0,082
Продуктивність (0,137)	Багатозадачність	0,035
	Обчислювальна потужність	0,014
	Точність	0,087
Вартість (0,077)	Розгортання	0,047
	Обслуговування	0,017
	Модернізація	0,013
Функціональність (0,161)	Технічна підтримка	0,022
	Навчання	0,059
	Представлення	0,080
Надійність (0,289)	Неінвазивність	0,199
	Значущість	0,054
	Відновлення	0,037
Масштабованість (0,060)	Інтегрованість	0,033
	Здатність до збільшення	0,014
	Мобільність	0,012

Як можна помітити, сума ваг критеріїв кожного рівня ієрархії дорівнює одиниці, а сума ваг підкритеріїв дорівнює значенню локального пріоритету критерію, якому вони належать.

1.4.4. Перевірка суджень на узгодженість

Перевірку на узгодженість проводимо за допомогою формул (3), (4) і (5), знаходимо лямбду, CI, CR. При цьому значення індексу послідовності (CR) співвідношень не повинно перевищувати 10% (20% - для систем, що не вимагають особливої точності). Провівши відповідні розрахунки за даними формулами, було знайдено індекси послідовності даних (CR) для матриць в табл.1.9-1.14.

Таблиця 1.16.

Індекси послідовності даних

Матриця	Лямбда	CI	CR
Інформативність	3,0092	0,0045	0,0079
Продуктивність	3,0385	0,0192	0,0331
Вартість	3,0941	0,0470	0,0810
Функціональність	3,0940	0,0470	0,0810
Надійність	3,0940	0,0470	0,0810
Масштабованість	3,0182	0,0091	0,0157

Як бачимо з табл.1.16 значення індексу послідовності CR не перевищує 10%.

1.4.5. Прийняття рішення на основі отриманих результатів

Останнім кроком в методі аналізу ієрархій буде побудова матриць, які будуть мати таку ж розмірність як і кількість альтернатив, в яких кожна з альтернатив буде порівнюватись між собою на основі критеріїв передостаннього рівня. В нашому випадку будуть порівнюватися дві телемедичні системи, тому розмірність матриць буде 2x2. В якості критеріїв будуть представлені 18 критеріїв

четвертого рівня. Після введення цих даних, буде здійснена їх обробка програмним продуктом та виведені результати кращої альтернативи з урахуванням всіх критеріїв, що були введені експертом.

На цьому кроці будуються матриці попарних порівнянь для альтернатив відносно кінцевих критеріїв ієрархії. В цих матрицях користувач (експерт) повинен вказати перевагу однієї телемедичної системи над іншою і степінь цієї переваги.

Таблиця 1.17.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Діагностичні ознаки»

Діагностичні ознаки	КМІС	Медіалог
КМІС	1	3
Медіалог	0,333	1

Вага альтернативи КМІС – 0,75.

Вага альтернативи Медіалог – 0,25.

Таблиця 1.18.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Сигнал»

Сигнал	КМІС	Медіалог
КМІС	1	1
Медіалог	1	1

Вага альтернативи КМІС – 0,5.

Вага альтернативи Медіалог – 0,5.

Таблиця 1.19.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Метод»

Метод	КМІС	Медіалог
КМІС	1	0,25
Медіалог	4	1

Вага альтернативи КМІС – 0,2.

Вага альтернативи Медіалог – 0,8.

Таблиця 1.20.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Багатозадачність»

Багатозадачність	КМІС	Медіалог
КМІС	1	1
Медіалог	1	1

Вага альтернативи КМІС – 0,5.

Вага альтернативи Медіалог – 0,5.

Таблиця 1.21.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Обчислювальна потужність»

Обчислювальна потужність	КМІС	Медіалог
КМІС	1	5
Медіалог	0,2	1

Вага альтернативи КМІС – 0,83.

Вага альтернативи Медіалог – 0,17.

Таблиця 1.22.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Точність»

Точність	КМІС	Медіалог
КМІС	1	3
Медіалог	0,333	1

Вага альтернативи КМІС – 0,75.

Вага альтернативи Медіалог – 0,25.

Таблиця 1.23.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Розгортання»

Розгортання	КМІС	Медіалог
КМІС	1	0,5
Медіалог	2	1

Вага альтернативи КМІС – 0,33.

Вага альтернативи Медіалог – 0,66.

Таблиця 1.24.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Обслуговування»

Обслуговування	КМІС	Медіалог
КМІС	1	0,333
Медіалог	3	1

Вага альтернативи КМІС – 0,25.

Вага альтернативи Медіалог – 0,75.

Таблиця 1.25.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Модернізація»

Модернізація	КМІС	Медіалог
КМІС	1	3
Медіалог	0,333	1

Вага альтернативи КМІС – 0,75.

Вага альтернативи Медіалог – 0,25.

Таблиця 1.26.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Технічна підтримка»

Технічна підтримка	КМІС	Медіалог
КМІС	1	5
Медіалог	0,2	1

Вага альтернативи КМІС – 0,83.

Вага альтернативи Медіалог – 0,17.

Таблиця 1.27.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Навчання»

Навчання	КМІС	Медіалог
КМІС	1	2
Медіалог	0,5	1

Вага альтернативи КМІС – 0,66.

Вага альтернативи Медіалог – 0,33.

Таблиця 1.28.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Представлення»

Представлення	КМІС	Медіалог
КМІС	1	3
Медіалог	0,333	1

Вага альтернативи КМІС – 0,75.

Вага альтернативи Медіалог – 0,25.

Таблиця 1.29.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Неінвазивність»

Неінвазивність	КМІС	Медіалог
КМІС	1	8
Медіалог	0,125	1

Вага альтернативи КМІС – 0,89.

Вага альтернативи Медіалог – 0,11.

Таблиця 1.30.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Захищеність»

Захищеність	КМІС	Медіалог
КМІС	1	0,333
Медіалог	3	1

Вага альтернативи КМІС – 0,25.

Вага альтернативи Медіалог – 0,75.

Таблиця 1.31.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Відновлення»

Відновлення	КМІС	Медіалог
КМІС	1	4
Медіалог	0,25	1

Таблиця 1.32.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Інтегрованість»

Інтегрованість	КМІС	Медіалог
КМІС	1	0,2
Медіалог	5	1

Вага альтернативи КМІС – 0,17.

Вага альтернативи Медіалог – 0,83.

Таблиця 1.33.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Здатність до збільшення»

Здатність до збільшення	КМІС	Медіалог
КМІС	1	1
Медіалог	1	1

Вага альтернативи КМІС – 0, 5.

Вага альтернативи Медіалог – 0,5.

Таблиця 1.34.

Матриця попарних порівнянь для вибору альтернатив за критерієм «Мобільність»

Мобільність	КМІС	Медіалог
КМІС	1	5
Медіалог	0,2	1

Вага альтернативи КМІС – 0,83.

Вага альтернативи Медіалог – 0,17.

Після введення останніх матриць попарних порівнянь, ми отримали значення ваг альтернатив відносно кожного з критеріїв. Таким чином можна буде обрахувати глобальні ваги альтернатив. Вага кожної альтернативи обчислюється як сума добутків глобальних пріоритетів критеріїв (табл.1.15) на вагу альтернативи відносно цих критеріїв (табл.1.17-1.34)

Глобальні значення альтернатив будуть мати наступні значення:

КМІС – 0,635496

Медіалог - 0,364504

Таким чином, можна зробити висновок, що телемедична система КМІС є кращою альтернативою з урахуванням всіх введених критеріїв.

1.5. Засоби для розробки програмного продукту

1.5.1. Вибір середовища розробки програмного забезпечення

Одним з ключових завдань дипломної роботи є вибір середовища розробки кінцевого програмного продукту. Потрібно вибрати таке середовище, в якому в повній мірі можна було б реалізувати програму, затративши якомога менше часу. При цьому програма повинна містити прозорий і зрозумілий іншим користувачам

код, та бути гнучкою на випадок змін. Також дуже важливо домогтися максимальної швидкодії та мінімізувати витрату ресурсів.

Враховуючи всі перераховані вище вимоги, було прийнято рішення використовувати платформу .NET і ASP.NET яка є складовою її частиною, для створення графічного інтерфейсу, який був би зручний та зрозумілий для користувача. Середовищем розробки під дану платформу було обрано продукт компанії Microsoft – Visual Studio 2012.

Ця платформа, завдяки готовому набору класів та об’єктів, дозволить написати програму максимально швидко, зберігши при цьому простоту, гнучкість, прозорість та надійність коду.

1.5.2. Платформа .NET

Платформа .NET була розроблена компанією Microsoft в 2002 році. Основу платформи складає загальномовна сфера виконання Common Language Runtime (CLR), яка підходить для різних мов програмування. Функціональні можливості CLR доступні на будь-яких мовах програмування, що використовують це середовище[8].

При проектуванні .NET Framework, компанія Microsoft врахувала всі недоліки існуючих Windows-платформ. .NET складається з двох частин:

- загальномовного середовища (Common Language Runtime, CLR);
- бібліотеки класів (Framework Class Library, FCL).

CLR надає модель програмування, що використовується в усіх типах прикладних програм. В CLR є власний завантажувач файлів, диспетчер пам’яті, система безпеки та ін. Крім того, CLR надає об’єктно-орієнтовану модель програмування, яка визначає, як виглядають і ведуть себе типи і об’єкти. FCL надає об’єктно-орієнтований API-інтерфейс, що використовується всіма моделями програм. В цій бібліотеці містяться визначення типів, які дозволяють розробникам

виконувати ввід/вивід, планування задач в інших потоках, створювати графічні образи та ін.

Нижче наведений список всіх переваг платформи .NET.

1. Повна і абсолютна взаємодія між мовами. В платформі .NET підтримується наслідування між різними мовами, обробка винятків і налагодження між мовами.

2. Загальне середовище виконання для всіх програм .NET незалежно від того, на яких мовах вони створені. Це пояснюється тим, що для всіх мов використовується один і той самий набір вбудованих типів даних.

3. Спрощена модель програмування. Вона допомагає користувачу уникати роботи з різними структурами, як в Win32 I COM. Відтак, розробнику не потрібно працювати з реєстром, глобальними унікальними ідентифікаторами (GUID), IUnknown, AddRef, HRESULT.

4. Відсутність проблем з версіями. Всі розробники, що працюють в ОС Windows часто мають справу з проблемами несумісності версій. Ця проблема виникає коли компоненти, шо були встановлені для нової програми, замінюють компоненти старої програми, і в результаті остання перестає працювати. Архітектура платформи .NET дозволяє ізолювати прикладні компоненти так, що програма завжди завантажує компоненти, з якими вона будувалась і тестувалась. Якщо програма працює після початкового встановлення, то це означає, що вона буде працювати надалі завжди.

5. Спрощене розвертання. Раніше Windows-додатки було дуже важко встановити і розвертати: потрібно було створити багато файлів, параметрів реєстру і ярликів. До того ж повністю видалити додаток практично неможливо. З появою .NET Framework все значно спростилося. Компоненти .NET не пов’язані з реєстром. Встановлення програм .NET зводиться до копіювання файлів в потрібні каталоги і створення ярликів. Видалення програм зводиться до видалення самих файлів.

6. Робота на багатьох платформах. При компіляції коду на .NET, компілятор генерує код на загальній проміжній мові (Common Intermediate Language. CIL), а не

традиційний код, що складається з процесорних команд. При виконання CIL транслюється в команди процесора. Оскільки трансляція виконується в період виконання, генеруються команди для конкретної моделі процесора. Це означає, що можна запускати свою програму, розроблену в .NET на будь якому комп'ютері, де підтримується .NET потрібної версії, що відповідає стандарту ECMA: з архітектурою x86, x64, IA64 та ін.

7. Інтеграція мов програмування. Технологія COM підтримує взаємодію різних мов - .NET Framework забезпечує інтеграцію різних мов, тобто одна мова може використовувати типи, створені на інших мовах. Наприклад, .NET Framework дозволяє створити на C++ клас, який походить від класу, реалізованого на VisualBasic. В CLR це можливо через наявність загальної системи типів (Common Type System, CTS), яку повинні використовувати всі мови, орієнтовані на CLR. Загальномовна специфікація (Common Language Specification, CLS) визначає правила, яких повинні дотримуватись розробники компіляторів, щоб їх мови інтегрувались з іншими. Сама Microsoft пропонує декілька таких мов: C++/CLI (C++ з керованими розширеннями), C#, VisualBasic NET. Крім того, інші компанії і навчальні заклади створюють компілятори інших мов, сумісних з CLR.

8. Спрощене повторне використання коду. Всі описані вище механізми дозволяють створити власні класи, що надають сервіс стороннім програмам. Тепер багатократне повторення коду є виключно простим і створюється велика база даних готових компонентів (типів).

9. Автоматичне керування пам'яттю (вибірка сміття). Програмування потребує значних навиків програміста, особливо коли йде мова про користування ресурсами (файлами, пам'яттю, простором екрану, мережевих з'єднань, ресурсів баз даних та ін.). Одна з найбільших помилок – нераціональне використання цих ресурсів, що може призвести до некоректного виконання програми в певний момент. CLR автоматично відслідковує використання ресурсів, гарантуючи цілісність даних та те, що не відбудеться витік інформації.

10. Перевірка безпеки типів CLR може перевіряти безпеку використання типів в коді, що гарантує коректне звернення до існуючих типів. Якщо вхідний

параметр методу оголошений як 4-байтне значення, CLR знайде і попередить передачу 8-байтного значення в якості значення цього параметру. Безпека типів також означає, що керування може передаватися тільки в певні точки (точки входу методів). Неможливо вказати будь-яку адресу та і заставити програму працювати з цієї точки. Сукупність всіх цих мір захисту допомагає не допустити багатьох поширених помилок при програмуванні (наприклад унеможливлює використання переповненого буферу обміну даних для «взлому» програми).

11. Розвинута підтримка налагоджень. Оскільки CLR використовується для багатьох мов, можна написати окремий фрагмент на мові, яка найбільше підходить для конкретної задачі, CLR повністю підтримує налагодження багатомовних програм.

12. Єдиний принцип обробки збоїв. Один з найбільш неприємних моментів Windows–програмування – це неузгоджений стиль повідомлень про збої. Одні функції повертають коди станів Win32, інші – HRESULT, інші – генерують винятки, які дозволяють відокремити код, необхідний для відновлення після себе, від основного алгоритму. Такий розподіл полегшує написання, читання і супроводження програм. Крім того, винятки працюють в багатомодульних і багатомовних програмах. На відміну від кодів станів і HRESULT винятки неможливо проігнорувати. CLR також надає вбудовані засоби аналізу стеку, що значно спрощують пошук фрагментів, що викликають збої.

Безпека. Традиційні системи безпеки забезпечують керування доступом на основі облікових записів користувачів. Це перевірена модель, але вона передбачає, що будь якому можна довіряти в однаковій степені. Таке припущення є оправданим коли весь код встановлюється з фізичних носіїв або з перевірених корпоративних серверів. Але по мірі збільшення об’єму мобільного коду, наприклад Web-сценаріїв, програм, що завантажуються з Інтернету, і даних, що містяться в електронній пошті, потрібен спосіб контролю за поведінкою програм, що орієнтований на сам код. Такий підхід реалізований в моделі безпеки доступу до коду.

- Взаємодія з існуючим кодом. В .NET Framework реалізована повна підтримка доступу до COM-компонентів і Win32-функцій в існуючих динамічних бібліотеках DLL.

На даний момент існують наступні версії .NET Framework:

Таблиця 1.35.

Список версій .NET Framework

Версія	Номер версії	Дата виходу	Visual Studio	Windows за замовчуванням
1.0	1.0.3705.0	1 травня 2002 року	Visual Studio .NET	
1.1	1.1.4322.573	1 квітня 2003 року	Visual Studio .NET 2003	Windows Server 2003
2.0	2.0.50727.42	11 червня 2005 року	Visual Studio 2005	Windows Vista, Windows 7, Windows Server 2008 R2
3.0	3.0.4506.30	6 листопада 2006 року	Visual Studio 2005 + extensions	Windows Vista, Windows Server 2008 Windows 7, Windows Server 2008 R2
3.5	3.5.21022.8	9 листопада 2007 року	Visual Studio 2008	Windows 7, Windows Server 2008 R2
4.0	4.0.30319.1	12 квітня 2010 року	Visual Studio 2010	Windows 8, Windows Server 2012
4.5	4.5.50709.17929	15 серпня 2012 року	Visual Studio 2012	Windows 8, Windows Server 2012

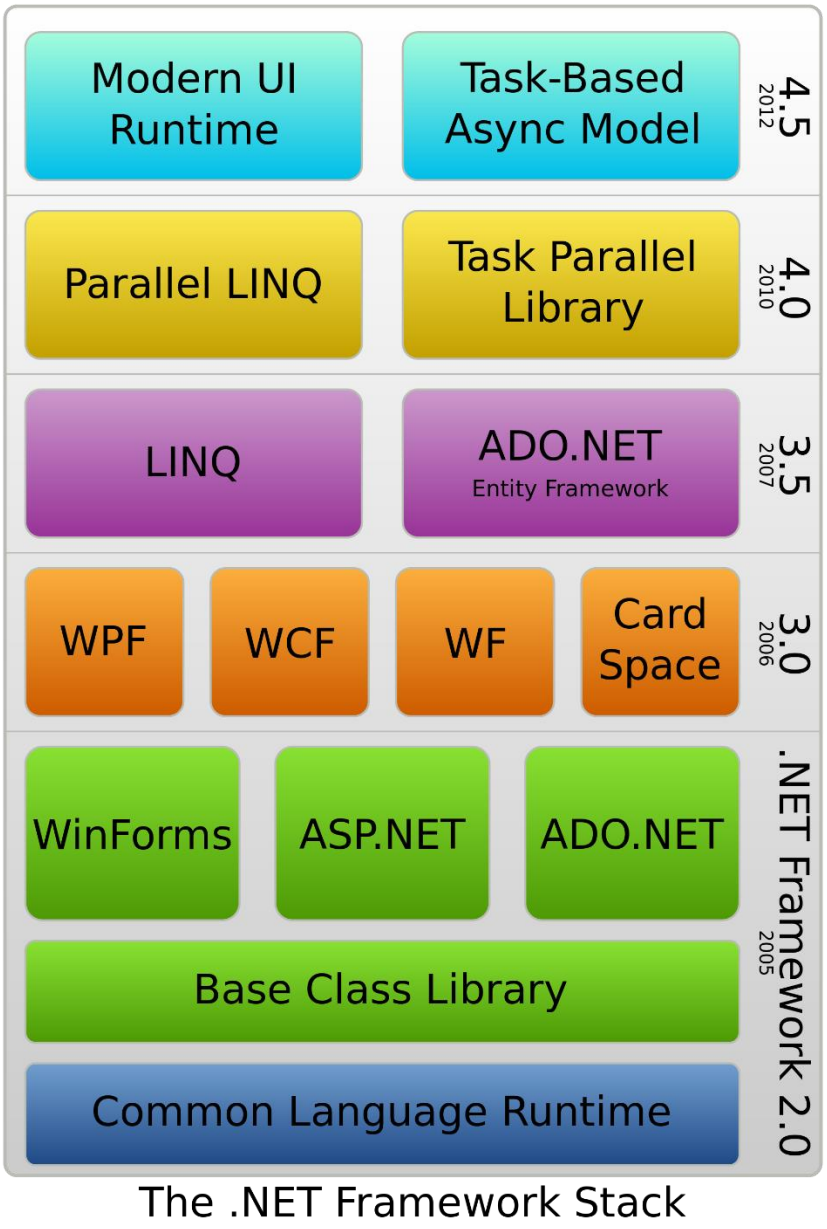

Рис.1.5. Стек технологій .NET Framework

.NET Framework підтримують такі середовища розробки:

- Microsoft Visual Studio;
- SharpDevelop;
- MonoDevelop;
- Embarcadero RAD Studio;
- Zonnon;
- PascalABC.NET.

В нашому випадку, для написання програмного продукту було обрано середовище розробки Microsoft Visual Studio 2012;

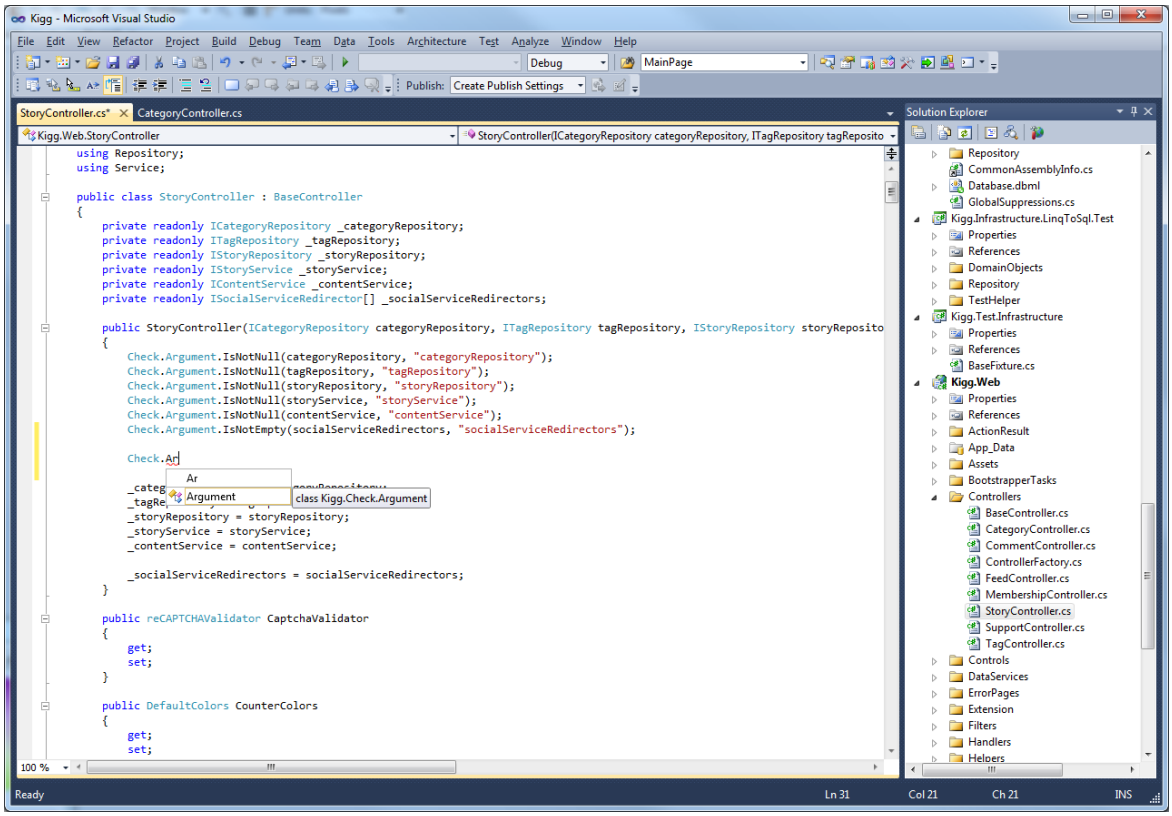

Рис.1.6. Вікно середовища розробки Visual Studio 2012

1.5.3. Платформа ASP.NET

Платформа ASP.NET являє собою технологію створення веб-застосунків від компанії Microsoft. Вона є складовою частиною платформи Microsoft .NET і розвитком більш старої технології Microsoft ASP. На даний момент останньою версією цієї технології є ASP.NET 4.5.

Компанія Microsoft повністю перебудувала ASP.NET, базуючись на Common Language Runtime (CLR), яка являє собою основу всіх застосунків Microsoft .NET. Розробники можуть писати код, використовуючи будь які мови програмування, що входять до набору Microsoft.NET (C#, Visual Basic.NET, Jscript.NET). Технологія ASP.NET має значну перевагу в швидкодії в порівнянні зі скриптовими технологіями[9].

Технологія ASP.NET має ряд переваг перед ASP.

1. Код, що компілюється, виконується швидше, більшість помилок знаходиться ще на стадії розробки.

2. Користувацькі елементи керування дозволяють виділяти шаблони, що часто використовуються, такі як меню сайту.

3. Розширений набір елементів керування і бібліотек класів дозволяє швидше розробляти програми.

4. Можливість кешування всієї сторінки або її частини для збільшення швидкодії.

5. Можливість кешування даних, що використовуються на сторінці.

6. Можливість розподілу візуальної частини і логіки за різними файлами.

7. Розширена модель обробки запитів.

8. Розширена модель подій.

9. Розширена модель серверних елементів керування.

10. Підтримка CRUD-операцій при роботі з таблицями через GridView.

11. Вбудована підтримка AJAX.

1.6. Програмна реалізація алгоритму

В даній дипломній роботі було розроблено програмне забезпечення для моделювання роботи методу аналізу ієрархії, в області телемедичних систем, шляхом порівняння двох телемедичних систем за розробленою ієрархією критеріїв. Ваги критеріїв повинні отримуватись з матриць попарних порівнянь, які в свою чергу повинні бути введені користувачем у відповідні таблиці через клавіатуру. Програма повинна також виводити проміжні дані, такі як значення локальних пріоритетів (ваги проміжних критеріїв), показувати індекс послідовності даних, який відображає адекватність та точність введених даних. Для завершення роботи програми, користувачу необхідно заповнити всі матриці попарних порівнянь для кожного з критеріїв. В кінцевому результаті програма виводить на екран глобальні

значення альтернатив, тим самим показуючи користувачу яка з альтернатив є кращою з урахуванням всіх введених даних.

Як було описано вище, програма розроблялася на мові програмування C#, для організації зручного інтерфейсу була використана платформа ASP.NET.

1.6.1 Принцип роботи і структура комп'ютерної програми

Загальний алгоритм роботи програми зображений на рис.1.7. На після запуску програми користувач вводить назви телемедичних систем та заповнює матрицю попарних порівнянь першого рівня. Обробка введених даних передбачає обрахування локальних пріоритетів, перевірку матриці на узгодженість, шляхом знаходження індексу послідовності даних, який буде виведений на екран. Далі структура є ідентичною: користувачу надаються форми для вводу матриць попарних порівнянь, а після їх введення обраховуються локальні (чи глобальні) пріоритети та індекс узгодженості. Коли користувач заповнить усі матриці, на екран будуть виведені глобальні значення альтернатив та програма запитає чи користувач бажає почати заново. У випадку натискання кнопки так – програма почне роботу спочатку, у випадку кнопки ні – програма завершиться.

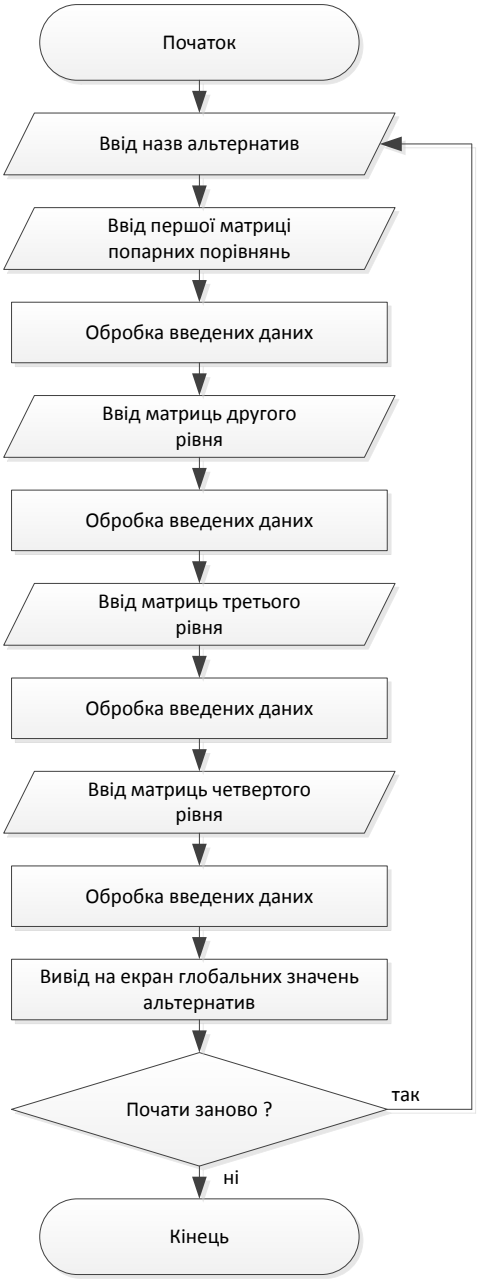

Рис.1.7. Загальний алгоритм роботи програми

Основними файлами програми є наступні.

1. Program.cs – ключовий файл в роботі програми. Містить точку входу в програму, що являє собою метод static void Main(), перелік бібліотек, необхідних для коректної роботи програми, запуск графічного інтерфейсу у вигляді форм а також текстові значення всіх компонентів програми.

2. DataContext.cs – файл, в якому описана логічна частина роботи програми. В ньому відбувається створення екземплярів класів, що являють собою матриці які в подальшому будуть вводитись, оголошені проміжні змінні, які будуть

використовуватися в процесі обрахунків, змінні, що містять кінцеві значення альтернатив, формули для їх обчислення а також зв'язок з графічним інтерфейсом.

3. Matrix_TA.cs – файл, що містить методи і змінні для роботи з матрицею попарних порівнянь першого рівня, у якій порівнюються такі критерії як «моніторинг», «передача даних», «діагностика», «збереження даних», «опрацювання даних» відносно цілі «вибір телемедичної системи». Виконання даного файлу передбачає знаходження ваг критеріїв та індексу послідовності даних (CR) для матриці першого рівня який в подальшому буде продемонстрований користувачу.

4. Matrix_AB.cs – файл, що містить методи і змінні для роботи з матрицею попарних порівнянь другого рівня, у якій порівнюються такі критерії як «інформативність», «продуктивність», «вартість», «функціональність», «надійність», «масштабованість» відносно таких критеріїв, як «моніторинг», «передача даних», «діагностика», «збереження даних», «опрацювання даних». Виконання даного файлу передбачає знаходження ваг критеріїв та індексу послідовності даних (CR) для матриці другого рівня.

5. Matrix_BBX.cs – файл, що містить методи і змінні для роботи з матрицею попарних порівнянь третього рівня, у якій порівнюються підкритерії «діагностичні ознаки», «сигнал», «метод», «багатозадачність», «обчислювальна потужність», «точність», «розгортання», «обслуговування», «модернізація», «технічна підтримка», «навчання», «представлення», «неінвазивність», «захищеність», «відновлення», «інтегрованість», «здатність до збільшення», «мобільність». Виконання даного файлу передбачає знаходження ваг критеріїв та індексу послідовності даних (CR) для матриці третього рівня.

6. Matrix_BXC.cs – файл, що містить методи і змінні для роботи з матрицею попарних порівнянь четвертого рівня, у якій порівнюються альтернативи відносно вищевказаних критеріїв. Виконання даного файлу передбачає знаходження ваг альтернатив. Індекс послідовності даних тут не обчислюється, оскільки для двовимірних матриць він завжди складає 0%.

7. Form1.cs – файл, що містить графічне представлення форми для вводу і виводу даних. Він містить два поля для вводу назв альтернатив а також матрицю 5x5 для вводу попарних порівнянь першого рівня.

8. Form2.cs – файл, що містить графічне представлення форми для вводу і виводу даних. Він містить 5 полів для вводу матриць 6x6. Також в ньому організований вивід значення індексу послідовності даних та значення локальних пріоритетів відповідної матриці.

9. Form3.cs – файл, що містить графічне представлення форми для вводу і виводу даних. Він містить 6 полів для вводу матриць 3x3. Також в ньому організований вивід значення індексу послідовності даних та значення локальних пріоритетів відповідної матриці.

10. Form4.cs – файл, що містить графічне представлення форми для вводу і виводу даних. Він містить 18 полів для вводу матриць 2x2. Також в ньому організований вивід значення індексу послідовності даних та значення локальних пріоритетів відповідної матриці.

11. Form5.cs – файл, що містить графічне представлення форми для вводу даних і виводу. У ньому організовано вивід на екран значень глобальних пріоритетів а також глобальні значення альтернатив. Крім того він надає можливість почати роботу програми заново або завершити програму.

Детальніше код програми можна розглянути в додатках до дипломної роботи.

1.6.2. Графічний інтерфейс програми

Графічний інтерфейс являє собою форми у які користувач може вводити свої дані та які повертають йому результати обрахунків, в тому числі і кінцеві значення альтернатив. Після заповнення форми користувач може перейти до наступного кроку. В загальному, інтерфейс програми має такий вигляд (рис.1.8).

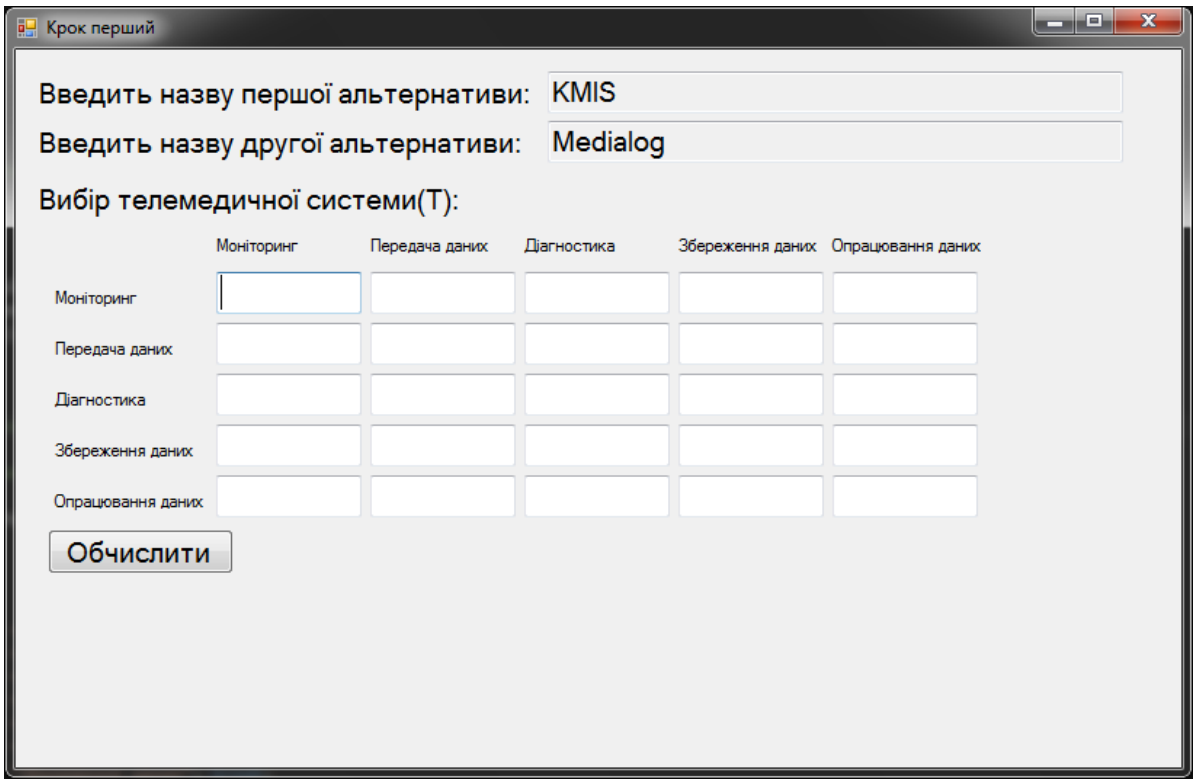

Рис.1.8. Вигляд форми для вводу альтернатив та матриць

Такий вигляд має форма для введення даних на першому кроці. Після введення назв альтернатив користувачу відкриваються поля для вводу матриці. Після заповнення цих полів, програма виведе на екран значення ваг цих критеріїв та індекс послідовності даних, після цього користувач зможе перейти до наступного кроку. Детальніше робота програми розглянути в спеціальній частині дипломної роботи.

РОЗДІЛ 2
СПЕЦІАЛЬНА ЧАСТИНА

В дипломній роботі розроблено програмне забезпечення, в основу якого покладено алгоритм методу аналізу ієрархій, для порівняння двох телемедичних систем. Цей алгоритм детально описано в розділі 1 п. 1.5. Програмний продукт реалізовано з використанням зручного графічного інтерфейсу з відповідними полями та формами для вводу та виводу даних. Результатом роботи програми є вивід на екран глобальних пріоритетів критеріїв та кращої альтернативи, отриманої в результаті обрахунків всіх критеріїв.

2.1. Вимоги до інформаційної та програмної сумісності програмного забезпечення

Для вхідних даних буде використано дані, введені з клавіатури, що являються собою попарні порівняння критеріїв між собою.

Надійність роботи програми забезпечується надійною роботою як апаратної частини, так і програмних засобів, включаючи операційну систему та власне програмний продукт.

Мінімальні вимоги до апаратної частини ЕОМ:

- процесори 1600 МГц і вище;
- об’єм оперативної пам’яті – 1,5 Гб.;
- місце на жорсткому диску - 10 Гб.;
- привід оптичних дисків (дисковод);
- пристрої вводу та виводу: клавіатура, миша, монітор.

Також для запуску програми необхідно встановити наступні компоненти:

- ОС Windows 7/8;
- .NET.Framework 4.5;
- Visual Studio 2012 або вищих версій, для відлагодження програми.

2.2. Верифікація програмного забезпечення

Функціональне призначення програми – моделювання роботи алгоритму методу аналізу ієрархій з використанням математичних обчислень, роботи з матрицями, на прикладі порівняння телемедичних систем.

Експлуатаційне призначення розробки – забезпечення обробки даних, введених у відповідні таблиці з клавіатури, виведення результатів обчислення на екран.

Основними перевагами розробленої комп'ютерної програми є:

- зручний користувацький інтерфейс;
- використання всієї обчислювальної потужності ПК;
- швидке опрацювання даних;
- виведення результатів на екран.

Швидкість роботи програми забезпечується потужним апаратним забезпеченням з використанням багатоядерних процесорів та великих об'ємів оперативної пам'яті. Програма тестувалась з використанням наступних програмних та апаратних даних:

- ОС Windows 7 Ultimate;
- .NET Framework 4.5;
- MS Visual Studio 2012;
- процесор Intel Core i3 M370 2.40 GHz;
- оперативна пам'ять 3,00 ГБ;
- 32-розрядна операційна система;
- відеокарта AMD Radeon HD 6550M.

Робота програми починається з запуску графічного інтерфейсу, розробленого за допомогою платформи ASP.NET, у якому користувачу пропонується введення назв альтернатив, в нашому випадку – телемедичних систем, що проходять порівняння. Далі користувачу пропонується ввести у відповідні таблиці користувацького інтерфейсу матриці попарних порівнянь заданих критеріїв для кожного рівня ієрархії, починаючи з найвищого і поки не буде заповнений останній

рівень, в якому містяться порівняння власне альтернатив. Після введення даних в кожну з матриць, користувачу буде виведено на екран значення локальних пріоритетів, а також такі індекси як: лямбда, індекс збіжності та індекс послідовності даних, які необхідні для перевірки введених даних на достовірність та коректність. Якщо ж індекс послідовності даних перевищує 10% - потрібно ввести дані ще раз, оскільки вони є недостатньо точними.

ЕОМ, яка використовувалася в програмі моделювання роботи методу аналізу ієрархій показала хорошу швидкодію. Час, необхідний для компіляції складає приблизно ~5 секунд, а час обробки даних для виводу результатів на екран є незначним і складає менше 1 секунди, оскільки процесор швидко виконує математичні операції, в даному випадку роботу з матрицями невеликих розмірностей.

2.3. Апробація програмного забезпечення

Після проведення тестувань, процес роботи програми можна умовно розділити на такі частини:

- ввід користувачем назв альтернатив;
- ввід матриць попарних порівнянь усіх рівнів;
- вивід на екран результатів обчислень.

Графічний інтерфейс представлений у вигляді п’яти форм:

- ввід користувачем назв альтернатив і матриці порівнянь першого рівня;
- ввід користувачем матриць попарних порівнянь другого рівня;
- ввід користувачем матриць попарних порівнянь третього рівня;
- ввід користувачем матриць попарних порівнянь четвертого рівня;
- вивід на екран результатів обчислень та значення альтернатив.

Далі представлені знімки виконання програми.

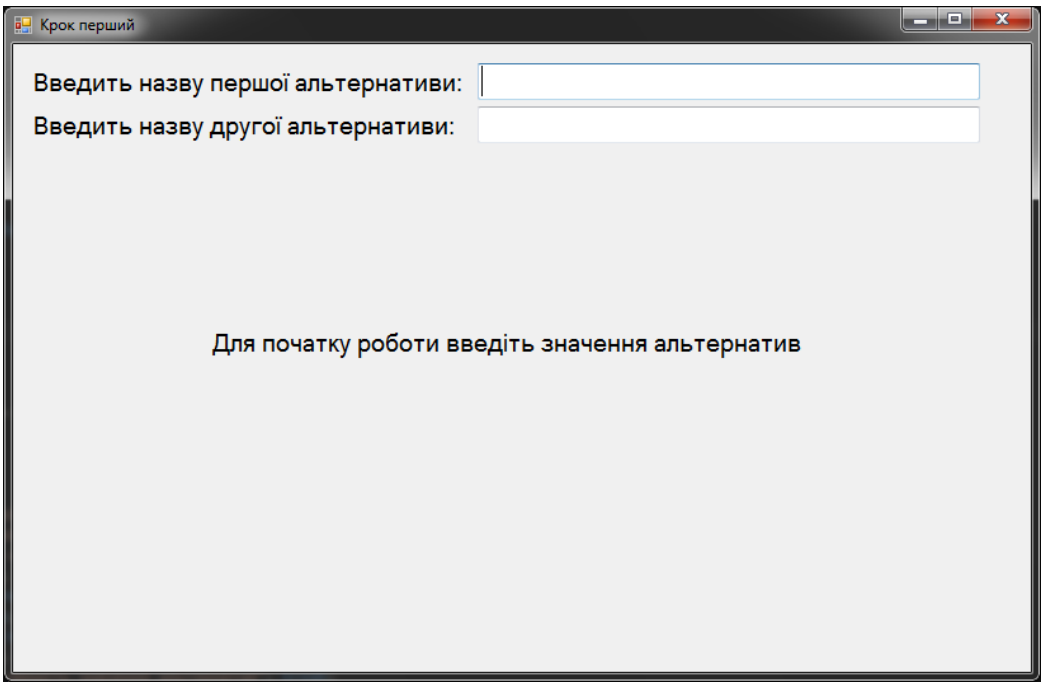

Рис.2.1. Вікно інтерфейсу: перший крок

Як бачимо з рис.2.1, користувач може ввести на власний розсуд імена альтернатив, які в подальшому будуть проходити порівняння, у відповідні поля. Після цього, натиснувши клавішу Tab, у цій же формі (форма 1) відкриється додаткове поле для вводу матриці попарних порівнянь першого рівня для основних критеріїв: моніторинг, передача даних, діагностика, збереження даних, опрацювання даних.

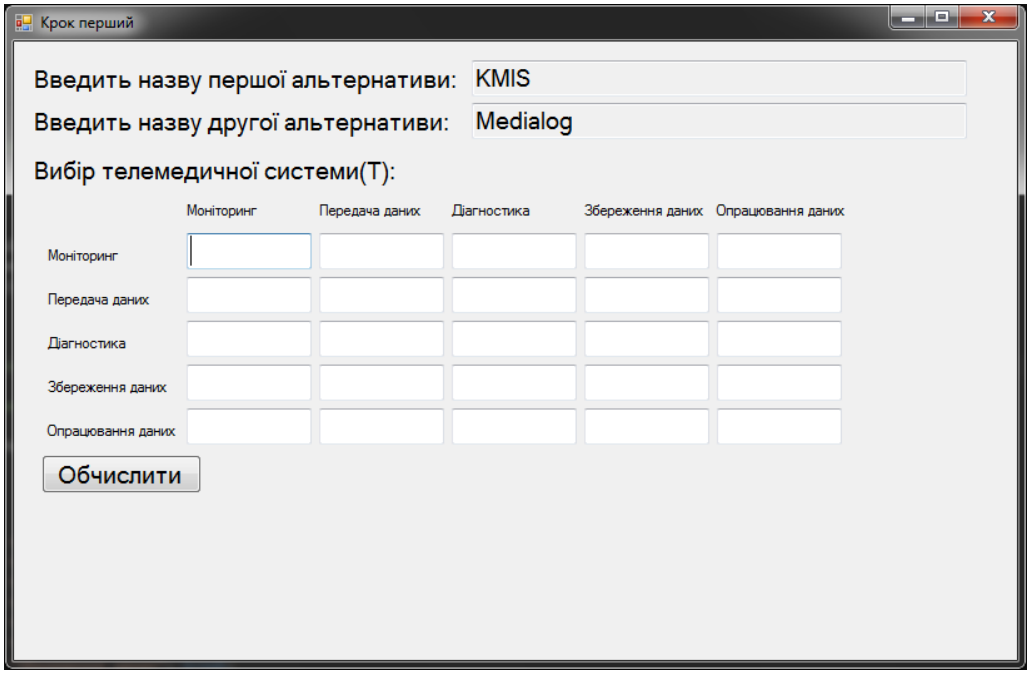

Рис.2.2. Вікно інтерфейсу: перший крок, ввід альтернатив

У це поле користувач вводить на власний розсуд ступені важливості у матрицю попарних порівнянь першого рівня.

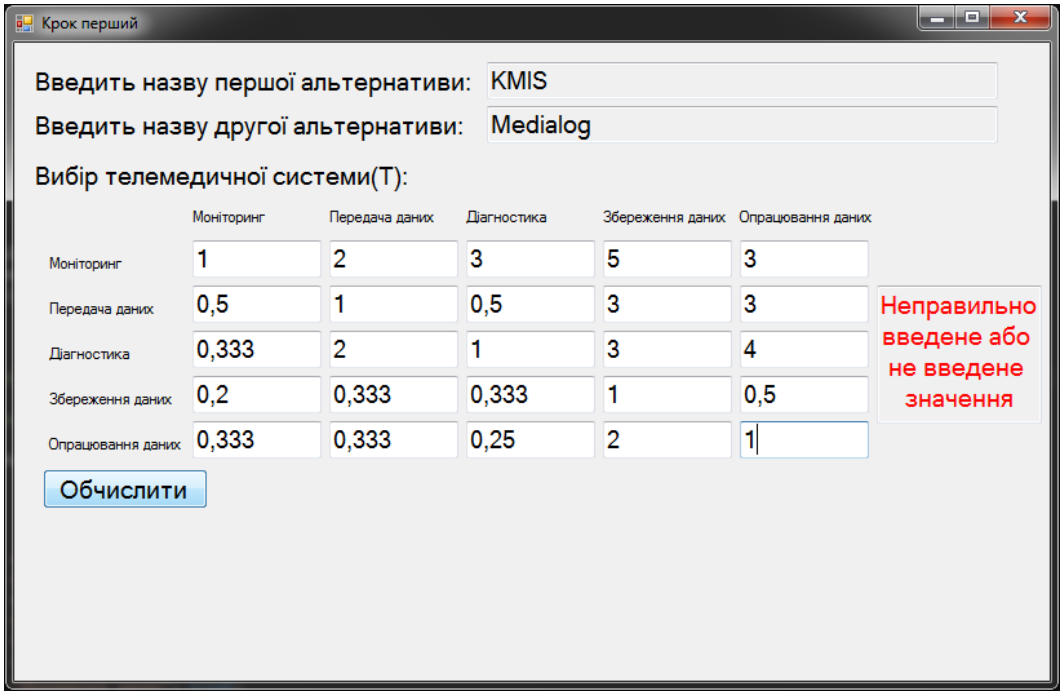

Рис.2.3. Вікно інтерфейсу: перший крок, ввід матриці

В результаті введених даних будуть отримані локальні пріоритети (ваги) цих критеріїв, а також дані будуть перевірені на точність і коректність, шляхом обчислення коефіцієнту послідовності даних.

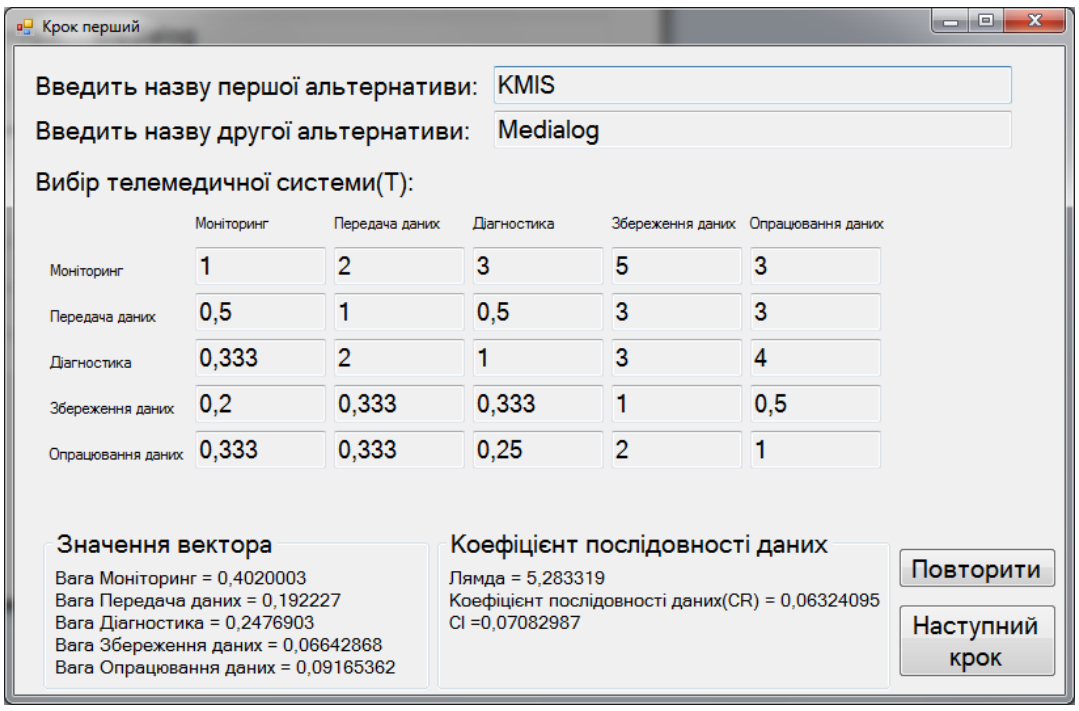

Рис.2.4. Вікно інтерфейсу: перший крок, завершення

Варто нагадати, що при вводі цих значень, ступінь важливості критеріїв що зліва порівнюються з критеріями зверху, а не навпаки. Як бачимо, коефіцієнт послідовності даних (CR) становить 0,0632, тобто приблизно 6%. Це означає, що дані введені досить точно, якщо ж цей коефіцієнт має значення більше 10%, користувач має можливість ввести дані заново, натиснувши клавішу «Повторити». Якщо ж дані введені правильно, як в нашому випадку, користувачу слід натиснути клавішу «Наступний крок» щоб перейти до другої форми, на якій він зможе ввести значення попарних порівнянь для матриці другого рівня.

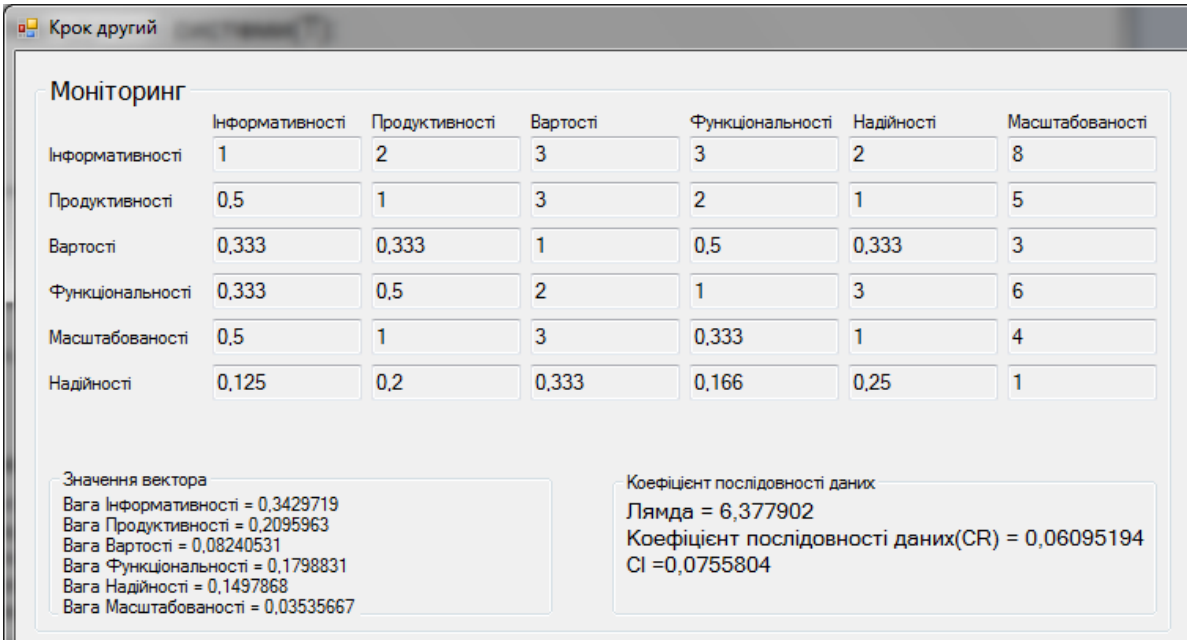

Рис.2.5. Вікно інтерфейсу: другий крок

В наступному кроці, користувач вводить матриці попарних порівнянь другого рівня, в якому кожен з шести критеріїв (інформативності, продуктивності, вартості, функціональності, надійності, масштабованості) буде порівнюватись між собою відносно кожного з п’яти критеріїв вищого рівня (моніторингу, передачі даних, діагностики, збереження даних, опрацювання даних). Відповідно таких матриць буде 5, оскільки у нас є 5 критеріїв вищого рівня. В результаті обчислень буде також виведено на екран локальні пріоритети (ваги) критеріїв та коефіцієнт послідовності даних.

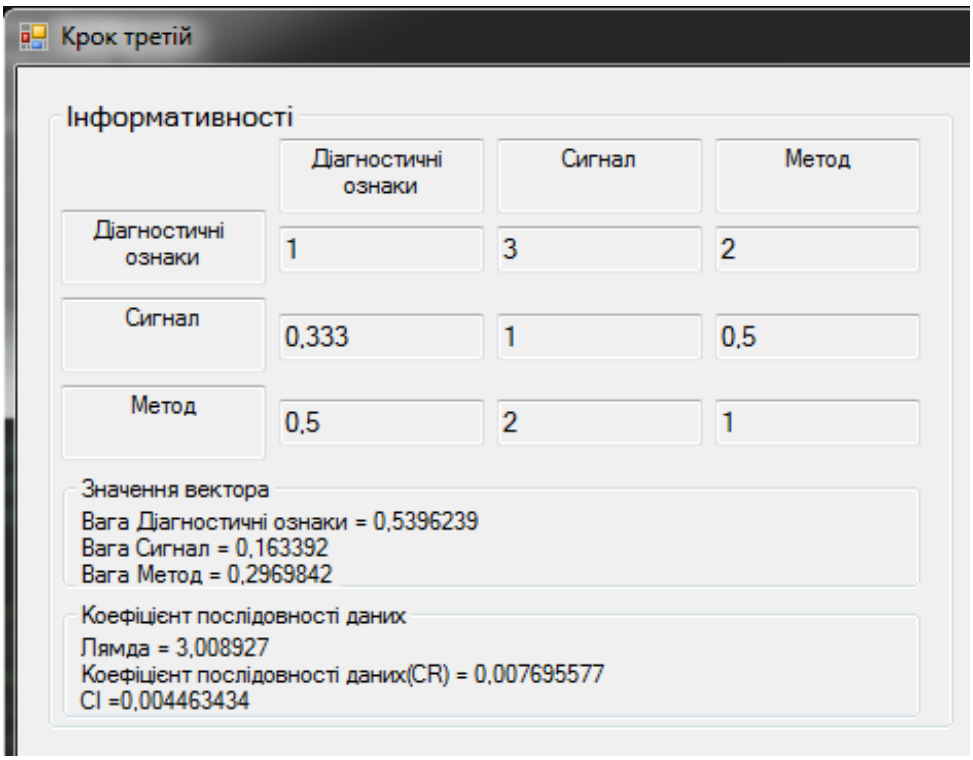

Рис.2.6. Вікно інтерфейсу: третій крок

В третьому кроці користувач вводить матриці попарних порівнянь для критеріїв четвертого рівня. Нагадаємо, що кожен з критеріїв третього рівня містить по 3 підкритерії четвертого рівня. Отримуємо ваги критеріїв та коефіцієнти послідовних значень. На рис. 2.6 показана матриця для підкритеріїв критерію інформативності. Таких матриць є 6 відповідно до кількості критеріїв.

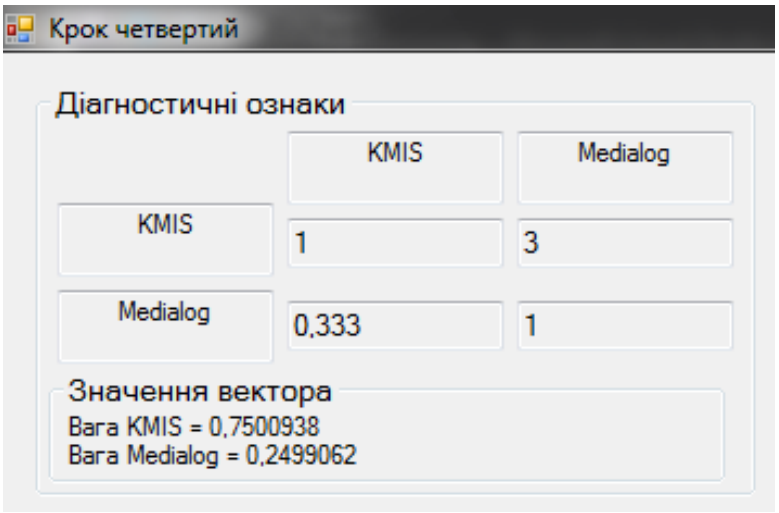

Рис.2.7. Вікно інтерфейсу: крок четвертий

В четвертому кроці користувач вводить матриці попарних порівнянь для кінцевих альтернатив відносно критеріїв, тим самим вказує яка телемедична система є кращою в даному аспекті і наскільки.

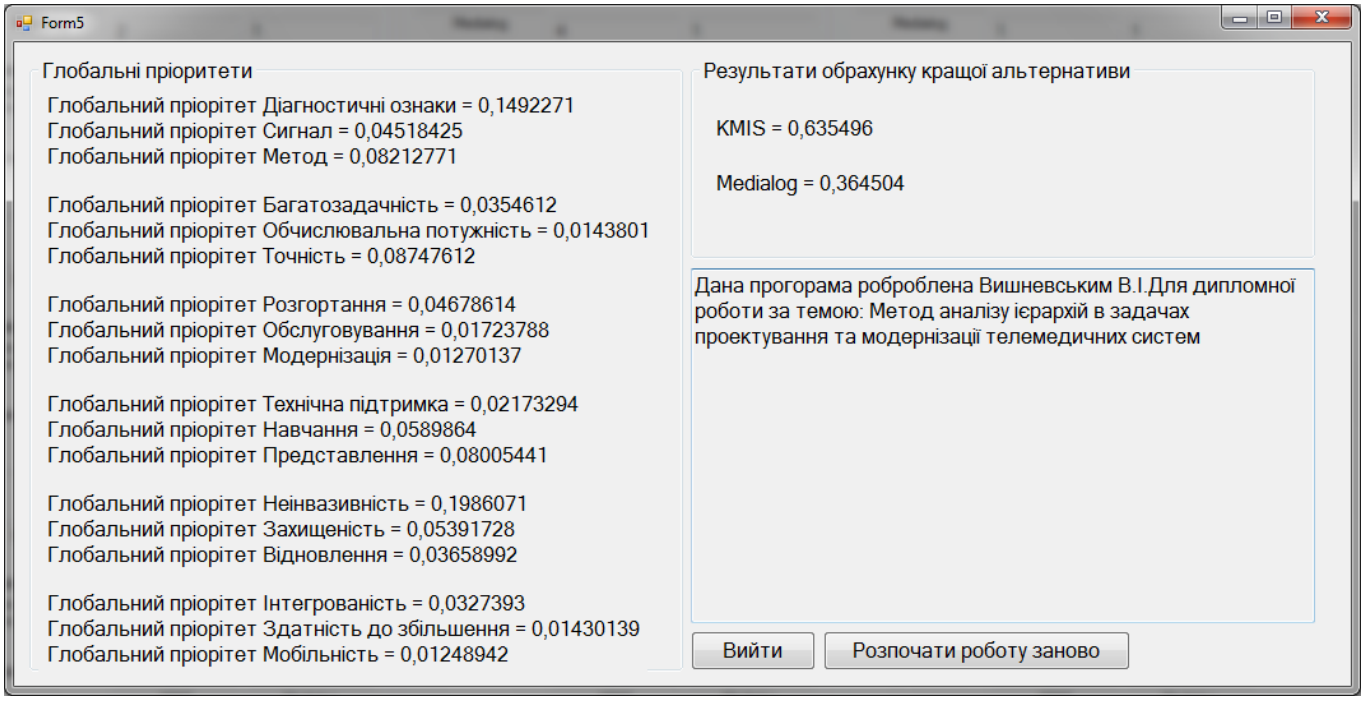

Рис.2.8. Вікно інтерфейсу: результат

Після введення всіх даних, можна побачити глобальні значення альтернатив, тобто яка альтернатива є кращою. Відповідно користувач має змогу вийти з програми натиснувши відповідну клавішу, або розпочати роботу заново.

РОЗДІЛ 3
ОБҐРУНТУВАННЯ ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ

Одним з основних етапів при здійсненні модернізації програмного забезпечення є розрахунок економічної ефективності від впровадження програмного продукту. Розробка повинна бути вигідною не тільки для розробника, а й для споживача, так як придбавши її він сподівається зменшити витрати часу, ресурсів. Отже необхідно оцінити економічний ефект для всіх сторін.

Метою дипломної роботи є розробка програми для моделювання роботи методу аналізу ієрархій в задачах проектування та модернізації телемедичних систем. В дану програму інформація вводиться з клавіатури особою, що приймає рішення, у відповідному графічному інтерфейсі. В кінцевому результаті обчислюється краща альтернатива з урахуванням всіх критеріїв.

3.1. Розрахунок норм часу на виконання проекту

Коефіцієнт корисної дії залежить від оптимального використання часу. Тому дуже велике значення має ефективне використання часу.

Основні етапи при створенні НДП:

- Постановка завдання.
- Огляд існуючих рішень при проектуванні програми.
- Аналіз сфери застосування.
- Проектування програми.
- Створення пакету документації.
- Оформлення документації.

Дані про витрати часу на виконання окремих етапів для визначення загальної тривалості проведення НДП представлені у табл.3.1.

Таблиця 3.1.

Основні етапи виконання НДП

№	Назва етапу	Середній час виконання етапу, год.	
		Інженер	Керівник
1	Постановка завдання	10	1
2	Огляд існуючих рішень при проектуванні програми	25	1
3	Аналіз сфери застосування	20	1
4	Проектування програми	62	3
5	Створення пакету документації	15	1
6	Оформлення документації	13	1
	Разом	145	8

Витрати часу керівника на виконання окремих етапів при недостатній кількості інформації доцільно приймати в межах 5% сумарних витрат часу інженерів на виконання цих етапів.

3.2. Розрахунок витрат на створення програмного забезпечення

Розрахунок поточних витрат на створення комп’ютерної програми проводять в розрізі таких калькуляційних статтей:

- основна заробітна плата (з/п);
- додаткова (з/п);
- нарахування на (з/п);
- консультаційні витрати;
- експериментально-виробничі витрати;

- загальновиробничі витрати;

- адміністративні витрати;

- позавиробничі витрати.

Системна обробка інформації щодо обліку праці і заробітної плати є однією з найрізноманітніших, складних і трудомістких ділянок роботи. Як показують дослідження за трудомісткістю ця частина становить майже одну третину від всього обліку по підприємству.

При розгляді питання про облік праці і заробітної плати та його системному розв'язанні велике значення має умовно-постійна (нормативна, довідкова та інша) інформація, яка в даному разі характеризує переважно постійних виконавців (людей і механізми) та постійні процеси (технологічні операції). Тому у першу чергу в зміст по обліку праці і заробітної плати неодмінно повинна входити інформація про виконавців (облік складу працівників).

Основна з/п складається із прямої з/п і доплати, яка при наближених розрахунках становить 25% – 35% від прямої з/п. При розрахунку з/п кількість робочих днів в місяці слід приймати – 25,4 дні/міс., що відповідає 203,2 год./міс. Нехай розмір місячного окладу керівника становить 5000 грн., а інженера 3000 грн., згідно існуючих на даний час норм.

Пряма з/п визначається:

$$ЗП = O_i * T_i / 203{,}2, \tag{3.1}$$

де O_i – розмір місячних окладів і-х категорій працівників;

T_i – трудомісткість робіт виконаних працівниками і-х категорій.

Для інженера: $ЗП = 3000 * 145 / 203{,}2 = 2140{,}75\, грн.$

Для керівника: $ЗП = 5000 * 8 / 203{,}2 = 196{,}85\ грн.$

Величина доплат обчислюється за формулою:

$$ЗП_1 = ЗП * K_i, \tag{3.2}$$

де $К_і$ –коефіцієнт доплат (0,25 - 0,35).

Нехай коефіцієнт становить 0,25:

Для інженера: $ЗП_1 = 2140,75 * 0,25 = 535,19 грн.$

Для керівника: $ЗП_1 = 196,85 * 0,25 = 49,21 грн.$

Основна з/п обчислюється за формулою:

$$ЗП_О = ЗП + ЗП_1 \quad (3.3)$$

Для інженера: $ЗП_О = 2140,75 + 535,19 = 2675,94 \ грн.$

Для керівника: $ЗП_О = 196,85 + 49,21 = 246,06 \ грн.$

Величина додаткової з/п обчислюється за формулою:

$$ЗП_Д = ЗП_О * К_Д, \quad (3.4)$$

де $К_Д$ – коефіцієнт додаткової з/п (0,05 - 0,1).

Нехай коефіцієнт додаткової $К_Д = 0,05$, тоді величина додаткової заробітної плати інженера становитиме:

$$ЗП_Д = 2675,94 * 0,05 = 133,79 \ грн.$$

Для керівника додаткова плата становить:

$$ЗП_Д = 246,06 * 0,05 = 12,3 грн.$$

Витрати на модернізацію комп'ютерної програми, крім річного фонду заробітної плати, включають ще й соціальні нарахування. Нормативи нарахувань на заробітну плату наступні:

- єдиний соціальний внесок – 3,6% ;

- прибутковий податок – 15% ;

Всього норматив нарахувань на заробітну плату становить 38% .

Загальний норматив нарахувань на заробітну плату розраховується за формулою:

$$З_Н = (ЗП_О + ЗП_Д) * К_Н, \quad (3.5)$$

де $ЗП_О$ – величина основної заробітної плати;

$ЗП_Д$ – величина додаткової заробітної плати;

$К_Н$ – загальний відсоток нарахувань на заробітну плату.

Для інженера загальний норматив нарахувань становить:

$$З_Н = (2675{,}94 + 133{,}79) * 0{,}38 = 1067{,}7 \ грн.$$

Для керівника загальний норматив нарахувань становить:

$$З_Н = (246{,}06 + 12{,}3) * 0{,}38 = 98{,}38 \ грн.$$

Зведена відомість витрат на заробітну плату представлена у табл.3.2.

Таблиця 3.2.

Зведена відомість витрат на заробітну плату, грн

№ п/п	Категорія працівників	Основна заробітна плата			Додаткова з/п	Нарахування з/п	Всього витрати на з/п
		Пряма з/п	Доплати	Всього:			
1	Інженер	2140,75	535,19	2675,94	133,79	1067,7	3877,43
2	Керівник	196,85	49,21	246,06	12,3	98,38	356,74
Разом		2337,6	584,4	2922	146,09	1166,08	4234,17

Якщо є необхідність, то при проведенні організаційно-економічного обґрунтування, слід враховувати витрати на консультації. Ці витрати можна порахувати окремою калькуляційною статтею виходячи з реальних цін на певний вид консультаційних послуг. Як правило, при отриманні консультацій, витрати рахуються на оплату праці консультантів за певний консультаційний час.

Щоб вивести розрахунок витрат на консультації, врахуємо, що консультації були надані в обсязі 8 год., вартість їх 356,74 грн.

Експериментально-виробничі витрати визначаються як витрати на машинний час, який є потрібним для виконання необхідного об'єму робіт виходячи з його вартості за одиницю часу, тобто:

$$З_{Е.В.} = B_P * T\,, \tag{3.6}$$

де $З_{Е.В.}$ – затрати експериментально-виробничі;

B_P – витрати на користування ПК та послуги інтернет;

T – час роботи ПК.

Вартість роботи на ПК і користування мережею Інтернет встановлюємо виходячи з реальних даних (14 грн./год.). Оскільки, інтернет та ПК використовувався на стадіях 2 - 6, то експериментально-виробничі затрати становлять:

$$З_{Е.В.} = 14*(25+20+62+15+13+1+1+3+1+1) = 1988 грн.$$

Загальновиробничі витрати при укрупнених розрахунках приймаємо на рівні 70% – 90% від суми основної і додаткової з/п інженерів, яка була нарахована за роботу при проведенні НДП , тобто:

$$З_{З.В.} = (ЗП_О + ЗП_Д) * К_З\,, \tag{3.7}$$

де $З_{З.В.}$ – загально-виробничі затрати;

$ЗП_О$ – основна заробітна плата;

$ЗП_Д$ – додаткова заробітна плата;

$К_З$ – коефіцієнт загальновиробничих затрат.

В даному випадку приймемо коефіцієнт загальновиробничих затрат на рівні 70%, тоді сума затрат становитиме:

$$З_{З.В.} = (2675{,}94 + 133{,}79) * 0{,}7 = 1966{,}81 \ грн.$$

Аналогічно визначаються адміністративні витрати, які доцільно приймати на рівні 50% - 60% від суми основної і додаткової з/п інженерів.

$$З_{З.А.} = (ЗП_О + ЗП_Д) * К_А, \quad (3.8)$$

де $З_{З.А.}$ – адміністративні витрати;

$ЗП_О$ – величина основної заробітної плати;

$ЗП_Д$ – величина додаткової заробітної плати;

$К_А$ – коефіцієнт адміністративних витрат.

Нехай коефіцієнт адміністративних витрат становить 50%, то величина адміністративних витрат буде рівна:

$$З_{З.А.} = (2675{,}94 + 133{,}79) * 0{,}5 = 1404{,}86 \ грн.$$

Витрати поза виробництвом слід приймати на рівні 3% – 7% від виробничої собівартості. Виробнича собівартість включає в себе основну заробітну плату, додаткову заробітну плату, нарахування на заробітну плату, консультації, експериментально-виробничі та загально-виробничі витрати, тобто:

$$В_{П.В.} = (ЗП_О + ЗП_Д + З_Н + М_З + З_{Е.В.} + З_{З.В.}) * К_{П.В.}, \quad (3.9)$$

де $В_{П.В.}$ – позавиробничі витрати;

$К_{П.В.}$ – коефіцієнт позавиробничих витрат.

В даному випадку нехай $К_{П.В.} = 3\%$, тоді витрати поза виробництвом становитимуть:

$$В_{П.В.} = (2675{,}94 + 133{,}79 + 1067{,}7 + 356{,}74 + 1988 + 1966{,}81) * 0{,}03 = 245{,}66 грн.$$

Розрахунок поточних витрати представлений в табл. 3.3.

Таблиця 3.3.

Калькуляція собівартості розробки програми

Статті витрат	Витрати, грн.	В % до загальної суми
Основна заробітна плата	2922	28,65%
Додаткова заробітна плата	146,09	1,43%
Нарахування на заробітну плату	1166,08	11,44%
Консультації	356,74	3,5%
Експериментально-виробничі витрати	1988	19,5%
Загальновиробничі витрати	1966,81	19,29%
Разом виробнича собівартість	8545,72	83,81%
Адміністративні витрати	1404,86	13,78%
Позавиробничі витрати	245,66	2,41%
Повна собівартість	10196,24	100%

3.3. Розрахунок ціни комп'ютерної програми і економічна ефективність від використання програмного продукту

Ціну програмного продукту можна визначити:

$$Ц = (C_{пр} / N + C_{коп}) + П, \qquad (3.10)$$

де $C_{пр}$ – собівартість програми, грн.;

N – кількість замовлень, од.;

$C_{коп}$ – собівартість копіювання (ксерокопії, дискети, компакт-диски, поштові витрати, відрядження спеціалістів для запуску та наладки програмного забезпечення тощо), грн.;

$П$ – нормативна величина прибутку (15% – 30% від собівартості $C_{пр}$).

Оцінка економічної ефективності розробки при проектуванні та налаштуванні програми становить:

$$Ц = (10196,24/1 + 1000) + 10196,24 * 0,15 = 12725,67 \ грн.$$

Для визначення ефективності продукту розраховують чисту приведену вартість NPV і термін окупності $T_{ОК}$:

$$NPV = -K + \sum_{i=1}^{t} \frac{П + A}{(1+i)^t}, \quad (3.11)$$

де K – капітальні вкладення, повна собівартість;

A – амортизація капітальних вкладень за період t (60% капітальних вкладень) ;

t – відповідний рік проекту;

i – дисконтна ставка (10% – 15%):

$$NPV = -10196{,}24 + \frac{12725{,}67 - 10196{,}24 + 12725{,}67 * 0{,}6}{1 + 0{,}15} +$$

$$+ \frac{12725{,}67 - 10196{,}24 + 12725{,}67 * 0{,}6}{(1 + 0{,}15)^2} = 6343{,}36$$

Термін окупності визначається за формулою:

$$T_{ОК} = Ц / \sum (NPV / (1+i)^t), \quad (3.12)$$

Термін окупності програмного забезпечення становить:

$$T_{ОК} = 12725{,}67 / (\frac{6343{,}36}{(1+0{,}15)} + \frac{6343{,}36}{(1+0{,}15)^2}) = 1{,}23 року.$$

Основні показники ефективності представлені в табл.3.4.

Таблиця.3.4

Основні показники ефективності.

№ п/п	Назва показника	Одиниці вимірювання	Величина
1	Витрати часу на розробку та організацію програмного продукту	год.	153
2	Витрати на розробку програмного забезпечення.	грн.	10196,24
3	Ціна програмного забезпечення	грн.	10172,64
4	Чиста приведена вартість програми	грн.	12725,67
5	Термін окупності витрат по проекту	рік	1,23

Проведені розрахунки свідчать про економічну доцільність розробки. Розробка вважається економічно ефективною, так як прийнятний термін окупності для нового програмного продукту становить T_o < 1…3 роки, а за нашими підрахунками становить 1,23 року.

РОЗДІЛ 4
ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1. Охорона праці

Охорона праці вивчає фактори виробничого середовища, організаційно-технічні і санітарно-гігієнічні умови, в яких здійснюється трудова діяльність людини, а також системи правових заходів щодо виконання правил техніки безпеки і виробничої санітарії.

Методологічною основою охорони праці є аналіз умов праці, технологічних процесів, виробничого обладнання, робочих місць, трудових операцій, організації виробництва з метою виявлення шкідливих і небезпечних факторів, виникнення можливих аварійних ситуацій та визначення заходів щодо поліпшення умов праці[16].

Основним завданням охорони праці є гуманізація праці, тобто профілактика перевтоми, професійних захворювань, запобігання виробничому травматизму, підвищення продуктивності праці, створювання умов для всебічного розвитку особистості. До завдань також входить знаходження оптимальних співвідношень між різними факторами виробничого середовища.

Робота з програмним забезпеченням працівниками телемедичних центрів, передбачає використання комп'ютерної техніки. Користувачі ЕОМ піддаються впливу шкідливих факторів. При виконанні робіт на персональному комп'ютері, користувач може піддаватися наступним небезпечним факторам:

- зорове навантаження;
- нервово-емоційне навантаження;
- відсутність чи недостатня кількість освітлення;
- недостатнє штучне освітлення робочої зони;
- підвищена яскравість світла;
- навантаження на суглоби;
- довготривала статична напруженість.

Для уникнення впливу шкідливих факторів при роботі з ПК, необхідно дотримуватися вимог згідно нормативних документів щодо забезпечення охорони праці користувачів ПК, а саме документу ДСанПіН 3.3.2.007–98 та НПАОП 0.00–1.28–10 «Державні санітарні правила й норми роботи з візуальними дисплейними терміналами (ВДТ) електронно-обчислювальних машин».

Умови робочого місця для роботи з програмою, призначеною для моделювання алгоритму методу аналізу ієрархій, визначаються:

- Умовами навколишнього середовища, а саме – освітлення в приміщенні телемедичного центру і на робочому місці, мікроклімат, шум, специфічні фактори, зумовлені особливостями засобів відображення інформації.

- Особливостями основних елементів робочого місця (просторові параметри робочого місця і його елементів, які повинні відповідати анатомо-фізіологічним даним працюючих).

- Характеристиками інформаційної взаємодії користувача і ПК.

Згідно вимог нормативних документів приміщення телемедичних центрів повинні забезпечувати площу 6,0м^2 на одне робоче місце та об’єм 20,0м^3 на одне робоче місце.

Також у приміщенні телемедичного центру повинно бути встановлено:

- опалення;

- систему кондиціонування повітря.

Також телемедичний центр повинен мати природне і штучне освітлення, яке б відповідало вимогам СНиП II-4-79, ДСанПіН 3.3.2.007-98.

Природне освітлення в приміщенні здійснюється через бічні світлові прорізи орієнтовані на північ. Коефіцієнт природної освітленості згідно з вимогами, повинен становити 1,5%. Віконні прорізи оснащені регульованими пристроями для відкривання- жалюзями.

Штучне освітлення в телемедичних центрах здійснюється системою загального рівномірного освітлення, яка включає суцільні або переривчасті лінії світильників, розташованих збоку робочих місць (ліворуч), паралельно лінії зору

користувачів ПК. Світильники оснащені розсіювачами світла та екрануючими сітками.

Допускається використання світильників наступних класів світлорозподілу:

- прямого світла – П;

- переважно прямого світла – Н;

- переважно відбитого світла – В.

Яскравість світильників загального освітлення, а також яскравість стелі при застосуванні системи відбитого освітлення не перевищує 200кд/м^2. Величина коефіцієнта пульсації освітленості не перевищує 5% - це забезпечується застосуванням газорозрядних ламп у світильниках загального і місцевого освітлення.

На робочих місцях користувачів ПК телемедичних центрів параметри мікроклімату повинні відповідати вимогам ДСН 3.3.6.042-99, ДСанПіН 3.3.2-007-98.

Для збереження здоров’я користувачів ПК, запобігання професійним захворюванням і підтримки працездатності при роботі з комп’ютерною програмою повинні бути дотримані вимоги ДСанПіН 3.3.2.007-98 щодо режиму праці та відпочинку. Для цього було призначено регламентовані перерви для відпочинку.

Протягом робочого дня передбачено:

- перерву для відпочинку і вживання їжі (обідня перерва);
- перерва для відпочинку і особистих потреб (згідно з трудовими нормами);
- додаткові перерви, а саме 15 хв., через кожну годину роботи.

При використанні комп’ютерної системи моделювання алгоритму роботи методу аналізу ієрархій повинні бути дотримані всі правила згідно з вимогами чинних нормативних документів ДСанПіН 3.3.2.007–98 та НПАОП 0.00–1.28–10, а саме:

- належні умови освітлення приміщення і робочого місця, відсутність відблисків;
- оптимальні параметри мікроклімату;

- належні ергономічні характеристики основних елементів робочого місця, а також враховувати небезпечні і шкідливі фактори, які були розглянуті раніше;
- перерви для відпочинку.

На основі наведених даних можна зробити висновок, що робота з комп'ютерною програмою, призначеною для моделювання роботи методу аналізу ієрархій в телемедичних системах, буде здійснюватися з дотриманням усіх правил охорони праці та санітарних норм, що стосуються експлуатації персональних комп'ютерів, освітлення робочих місць в телемедичних центрах чи інших медичних установах, створення відповідного мікроклімату в цих установах.

4.2. Захист ІТП від впливу хімічно-небезпечних речовин

Надзвичайна ситуація – це порушення нормальних умов життя і діяльності людей на об’єкті або території, спричинене аварією, катастрофою, стихійним лихом, епідемією, епізоотією, масштабною пожежею, застосуванням засобів ураження, що призвели або можуть призвести до людських і матеріальних втрат. Надзвичайні ситуації можуть поділятися за такими основними ознаками, як сфера виникнення, галузева ознака та за масштабом можливих наслідків.

Протягом свого життя інженери часто стикаються з великою кількістю шкідливих речовин, які можуть викликати різні види захворювань, розлади здоров’я, а також травми як у момент контакту, так і через певний проміжок часу. Особливу небезпеку становлять хімічні речовини, які залежно від їх виробничого використання можна поділити на:

- промислові отрути, які використовуються у виробництві (розчинників, барвників). Вони є джерелом небезпеки гострих і хронічних інтоксикацій (ртуть, свинець, ароматичні сполуки, тощо);

- отрутохімікати, що використовуються у сільському господарстві для боротьби з бур’янами та гризунами (гербіциди, пестициди);

- лікарські препарати;

- хімічні речовини побуту (харчові добавки, засоби санітарії та гігієни, косметичні засоби);

- хімічна зброя.

Залежно від характеру дії на організм людини хімічні речовини поділяються на: токсичні, подразнюючі, мутагенні, канцерогенні, наркотичні та задушливі.

Токсичні речовини – це речовини, які викликають отруєння всього організму людини, або впливають на окремі системи людського організму (кровотворну, центральну нервову). Ці речовини викликають патологічні зміни певних органів, наприклад нирок, печінки. До таких речовин належать такі сполуки, як чадний газ, селітра, розчини кислот та лугів.

Подразнюючі речовини викликають подразнення слизових оболонок, дихальних шляхів, очей, легень, шкіри (пари кислот, лугів, аміак).

Мутагенні речовини призводять до порушення генетичного коду, зміни спадкової інформації (свинець, радіоактивні речовини).

Канцерогенні речовини викликають, як правило, злоякісні новоутворення, такі як пухлини (ароматичні вуглеводні циклічні аміни, азбест, нікель, хром).

Наркотичні речовини впливають на центральну нервову систему (спирти, ароматичні вуглеводні).

Задушливі речовини призводять до токсичного набряку легень (оксид вуглецю, оксиди азоту).

Найбільш негативні наслідки має вплив саме отруйних речовин на живі організми, повітря, ґрунт, воду. Отруйними речовинами називаються ті, що призводять до ураження всіх живих організмів, особливо людей і тварин.

За вибірковістю дії шкідливі речовини можна поділити на:

1. Серцеві – кардіотоксична дія (ліки, рослинні отрути, солі кобальту).

2. Нервові – порушення психічної активності (чадний газ, фосфорорганічні сполуки, наркотичні засоби, снотворні ліки).

3. Печінкові – хлоровані вуглеводні альдегіди, феноли, отруйні гриби.

4. Ниркові – сполуки важких металів, етиленгліколі, щавлева кислота.

5. Кров’яні – анілін, нітрити.

6. Легеневі – оксиди азоту, озон, фосген.

За тривалістю дії шкідливі речовини ділять на три групи:

1. Летальні, що призводять до смерті (5% випадків) – термін дії до 10 діб.

2. Тимчасові, що призводять до нудоти, блювоти, набряку легень, болю в грудях – термін дії від 2 до 5 діб.

3. Короткочасні – тривалість декілька годин (подразнення у носі, ротовій порожнині, головний біль, задуха, загальна слабкість).

Певні види АХОВ (аварійно хімічно небезпечна речовина) перебувають у підприємствах які їх виробляють чи використовують. Великі запаси цих речовин мають підприємства хімічної, целюлозно-паперової, оборонної, та нафтохімічної

промисловості. Висока швидкість формування та дії вражаючих чинників АХОВ викликають необхідність прийняття оперативних заходів захисту персоналу, що можуть зазнати їх впливу. Захист від АХОВ є комплексом заходів, здійснюваних з метою винятку чи максимального ослаблення ураження персоналу та населення, збереження їхньої працездатності. Комплекс заходів щодо захисту від АХОВ включає:

- інженерно-технічні заходи щодо правильного зберігання, транспортування та використання АХОВ;

- підготовку зусиль і коштів на ліквідацію хімічно-небезпечних аварій;

- навчання ладу і правил поведінки за умов виникнення аварії персоналу об’єктів та населення;

- забезпечення засобами індивідуального захисту та колективного захисту;

- повсякденний хімічний контроль;

- прогнозування зон можливого хімічного зараження;

- попередження (оповіщення) про безпосередню загрозу ураження АХОВ;

- хімічну розвідку району аварії;

- тимчасову евакуацію персоналу об’єктів та населення з небезпечних районів;

- пошук постраждалих людей і надання їм допомоги;

- локалізацію і ліквідацію наслідків аварій.

Захист від АХОВ організовується та здійснюється передусім, безпосередньо на хімічно-небезпечних об’єктах. Ці заходи позначаються завчасно на плані захисту персоналу від АХОВ, що розроблюється завчасно.

Усі заходи щодо захисту інженерно-технічних працівників визначаються «Планом дій із попередження та ліквідації надзвичайних ситуацій», що є керівним документом з виконання заходів у випадку загрози і виникненні надзвичайних ситуацій. У ньому викладаються:

- прогноз і оцінка можливої обстановки у разі виникнення надзвичайних ситуацій, ступінь їхнього впливу, наслідки для людей, навколишнє середовище;

- обсяг, строки і порядок виконання заходів із попередження чи зниження небезпечних наслідків надзвичайних ситуацій;

- обсяг, строки і порядок виконання заходів щодо захисту населення чи працівників підприємств;

- склад угрупування сил, виділених на проведення аварійно-рятувальних, аварійно-відбудовних та інших невідкладних робіт, порядок приведення в готовність застосування;

- склад відомчих зусиль і коштів, залучуваних до робіт на ліквідацію, строки їхньої готовності і порядок здійснення маневрів ними на ході проведення цих робіт.

Висновок

Згідно з тематикою дипломної роботи магістра, для забезпечення роботи працівників, що використовують програмне забезпечення, призначене для моделювання роботи методу аналізу ієрархій, потрібно забезпечити їх захист від аварійно небезпечних хімічних речовин. У разі ж виникнення надзвичайних ситуацій, пов’язаних з викидом в медичній установі чи іншому об’єкті хімічно-небезпечних речовин, потрібно провести комплекс заходів, спрямований на захист працівників від цих речовин, та забезпечити всі умови для збереження їх життя та здоров’я. Також потрібно проводити роботи, пов’язані з наданням інформації працівникам про хімічно-небезпечні речовини, їх вплив на організм, та способи захисту від них.

РОЗДІЛ 5
ЕКОЛОГІЯ

5.1. Стратегія і тактика збереження й розвитку життя на землі

На Конференції ООН з навколишнього середовища і розвитку в Ріо-де-Жанейро (1992) були сформульовані принципи про нерозривність еколого-економічних зв'язків: економічний розвиток у відриві від екології приводить до перетворення планети на пустелю; натиск на екологію без економічного розвитку закріплює злидні і несправедливість.

Особливий акцент ставився на тому, що поняття сталого розвитку суспільства припускає забезпечення можливості задоволення потреб нинішнього покоління, людей без загрози можливості задоволення потреб поколінь прийдешніх. У широкому розумінні стратегія екорозвитку спрямована на досягнення гармонії між людьми (один з одним), між суспільством і природою.

Стратегія екорозвитку базується на таких основних принципах:

- забезпечення гармонізації співіснування людини і природи;

- невід'ємність захисту навколишнього середовища в процесі розвитку суспільства;

- відповідальність держави за погіршення стану навколишнього середовища в процесі розвитку суспільства;

- нарощування національного потенціалу країни для забезпечення екорозвитку;

- здійснення заходів щодо екологізації господарської діяльності, усунення причин забруднення, а не їх наслідків;

- проведення оцінки екологічних наслідків усіх видів діяльності, які можуть негативно вплинути на навколишнє природне середовище;

- регіональні і локальні завдання екорозвитку повинні підпорядковуватися глобальним і національним цілям запобігання екологічної кризи й оптимізації середовища існування людини (принцип «мислити глобально - діяти локально»);

- регіональний екорозвиток передбачає функцію раннього запобігання несприятливим екологічним тенденціям або ж гарантії їх мінімізації (незнання наслідків не звільнює суспільство від відповідальності за руйнацію навколишнього середовища);

- цілі екорозвитку є первинним щодо цілей економічного розвитку (принцип екологічного імперативу);

- розміщення і розвиток матеріального виробництва на певній території необхідно здійснювати відповідно до її екологічної техноємності (принцип еколого-економічної збалансованості);

- екологічна безпека суспільства тісно пов'язана з рівнем культури і вихованості людей в цьому суспільстві.

З цією метою було прийнято документ «Порядок денний на XXI століття» на наступні 100 років. Ця програма прагне досягти у глобальному масштабі двох цілей: високої якості довкілля та стабільної економіки для всіх народів світу.

Було також прийнято декларацію з довкілля та розвитку, які містить такі основні принципи:

- кожен має право на здорове та продуктивне життя у гармонії з природою;

- теперішнє та майбутні покоління мають рівні права на це;

- охорона довкілля має розглядатись як невід’ємна частина будь-якого процесу розвитку;

- кожна країна має право використовувати власні ресурси, не здійснюючи вплив на довкілля за межами її території;

- забруднювач повинен відшкодовувати збитки, завдані довкіллю (принцип «забруднювач платить»);

- економічна діяльність має бути поєднана з принципом реалізації запобіжних заходів у сфері охорони довкілля;

- держави повинні співпрацювати у сфері охорони довкілля;

- скорочення масштабів бідності та несправедливих стандартів життя в різних частинах світу є інтегральною частиною сталого (збалансованого) розвитку;

- держави повинні обмежувати та усувати моделі незбалансованого виробництва і споживання та сприяти впровадженню відповідної демографічної політики;

- найефективніший спосіб розв’язання екологічних проблем – залучення всіх зацікавлених осіб. Держави повинні розвивати та заохочувати поінформованість населення та його участь в процесі прийняття рішень;

- держави повинні розробити та впроваджувати ефективне законодавство у сфері охорони довкілля;

- до охорони довкілля мають залучатись усі соціальні групи;

- мир, розвиток і охорона довкілля є взаємопов’язаними та нероздільними.

5.2. Програмне забезпечення еколого-статистичних досліджень

На сьогоднішній день оперативна, якісна і точна обробка великих масивів даних і статистичної інформації може бути виконана лише з використанням сучасних засобів обчислювальної техніки. Наявність потужних, надійних і разом з тим простих в експлуатації програмних продуктів статистичного аналізу звільняє дослідника від рутинних операцій, розширює сферу застосування статистичних методів в різних галузях людської діяльності, сприяє появі якісно нових можливостей статистичного аналізу і моделювання даних. Використання пакетів прикладних програм - це єдиний реальний практичний інструмент розв'язування задач багатофакторного кореляційно-регресійного та аналізу в багатовимірному просторі.

Програмне забезпечення статистичних досліджень досить розвинуте. Сучасний ринок програмних продуктів пропонує різноманітні пакети програм для статистичної обробки даних. Всесвітньо відомі статистичні пакети для комплексної обробки даних: BMDP, SPSS, SAS, Systat, Minitab, S-Plus, Statgraphics Statistica та інші.

Використання згаданих пакетів програм дає змогу автоматизувати процес статистичного дослідження в таких напрямах:

- створення файлів даних і таблиць;
- групування даних;
- графічний аналіз даних;
- розрахунок варіаційних характеристик вибіркових сукупностей;
- побудова рядів розподілу;
- аналіз рядів динаміки і прогнозування їх майбутніх рівнів;
- кореляційно-регресійний аналіз;
- багатомірний аналіз.

З 1995 р. Світовим лідером на ринку статистичного програмного забезпечення визнається інтегрована система Statistical для Windows (версія 5.0), розроблена фірмою Stat Soft. Перша версія програми з'явилася у 1991р. для операційної системи MS-DOS і була новим напрямом розвитку статистичного програмного забезпечення. В ній реалізовано графічно-орієнтований підхід до статистичного аналізу даних, суть якого полягає в отриманні всебічного візуального представлення інформації на всіх етапах статистичної обробки даних.

Багатофункціональна, графічно орієнтована на обробку масових даних система Statistica відповідає основним стандартам Windows (динамічний обмін даними з іншими додатками, підтримка основних операцій з буфером обміну, робота в мережевому середовищі та інші).

Передусім це стандарти користувацького інтерфейсу — MDI, використання буфера-обміну, механізму динамічного зв'язку (DDE) з іншими додатками; система підтримує всі операції, реалізовані за допомогою методу Drag-and-Drop — «Перетягти та опустити», включаючи автозаповнення, інші.

Складніші процедури обробки даних у системі Stratgraphics виконує спеціалізований модуль Data Management — «Управління даними», а для обробки великих масивів даних або даних з довгими текстовими значеннями застосовують процедури Megafile Manager Data — «Менеджера метафайлів».

Система Stratgraphics працює з чотирма типами документів:

1) Електронна таблиця Spreadsheet, призначена для введення і перетворення первинних даних.

2) Електронна таблиця Scrollsheet — для виведення результатів аналізу.

3) Графік — для візуалізації результатів обробки та аналізу даних.

4) Звіт — файл у формі RTF (розширений текстовий формат), в якому зберігається текстова, числова і графічна інформація.

Усі статистичні процедури системи розбито на окремі модулі, кожен з яких об'єднує групу логічно зв'язаних між собою статистичних методів і в рамках конкретної моделі забезпечує повний і всебічний аналіз закономірностей.

У системі Statistica реалізовано принцип постійного логічного підказування. Якщо користувач не може визначитися щодо наступного кроку діалогу, через команду Enter система сама спрямує до відповідного діалогового вікна. Якщо виникають складнощі з вибором параметрів обчислювальної процедури, вони задаються системою «за умовчуванням».

Важливою характеристикою системи с наявність засобів всебічної графічної підтримки процесу обробки даних і візуалізації результатів аналізу. Графічні можливості й засоби системи унікальні. Вона включає сотні різних типів користувацьких і спеціальних статистичних графіків, доступних у будь-якому модулі й на будь-якому етапі статистичної обробки даних. Інструменти компонування складної графічної інформації з текстовою і числовою інформацією розглядаються у кожному модулі.

Використання сучасних комп'ютерних технологій обробки даних, інтерактивний спосіб взаємодії з системою перетворюють статистичний аналіз, моделювання та прогнозування в захоплююче дослідження закономірностей навколишнього світу. Завдяки різноманітним формам організації діалогу, максимально простій із звичними для статистики термінами мові спілкування, наявності контекстно-залежної довідкової системи, мові програмування STATISTICA BASIC пакет є ефективним інструментом проведення статистичного дослідження як для користувача-початківця, так і для професіонала.

5.3. Статистика природних та екологічних чинників

Сили, що діють з боку навколишнього середовища називають факторами. Розрізняють багато видів факторів, серед яких найбільш важливими для екологічної статистки є природні, соціальні та екологічні.

Природний фактор – це будь-який фактор (предмет, явище, рушійна сила процесів, умови їх перебігу), що діє незалежно від людини та без її участі або пов'язаний з її біологічною сутністю.

Соціальний фактор – це фактор, що є результатом функціонування людського суспільства.

Екологічний фактор – це будь-який елемент середовища, який здатний справляти прямий чи опосередкований вплив на живі організми, хоча б протягом однієї фази їхнього розвитку.

До 1980 року традиційним підходом до класифікації природних факторів був їх розподіл на природні ресурси та природні умови. Під природніми ресурсами традиційно розуміють тіла і сили природи, що на даному рівні розвитку продуктивних сил можуть бути використані в суспільному виробництві. Під природніми умовами розуміють тіла і сили природи, які мають істотне значення для життя і діяльності людського суспільства, однак безпосередньо або побічно не залучені до сфери виробничої діяльності людей (клімат, космічні промені та ін.). Принципово новий підхід запропонував М.Ф.Реймерс в 1994 році, його концепція базується на наявності так званого інтегрального ресурсу, що розглядається як системне утворення, яке експлуатується різними сільськогосподарськими галузями і підтримує життя на Землі. Більш ніж 76 компонентів, які входять до нього, утворюють інтегральні й комплексні сукупності. До них входять такі групи:

- Енергетичні ресурси (разом 16 одиниць).
- Газово-атмосферні (6 одиниць).
- Водні (11 одиниць).
- Ґрунтово-геологічні (11 одиниць).
- Біологічні (рослини, тварини, мікроорганізми 19 одиниць).

- Кліматичні (2 одиниці).

- Рекреаційні ресурси (3 одиниці).

- Антропоекологічні (3 одиниці).

- Інформаційні (2 одиниці).

- Ресурси простору і часу (3 одиниці).

Також ресурси можна класифікувати як:

- Вичерпні і невичерпні – ресурси, що з часом виснажуються в ході їх економічного використання (ґрунт, ліс, дикі тварини, кормові угіддя, копалини та ін.), і ті ресурси, зміни яких прямо не пов’язані з інтенсивністю їх використання (сонячна енергія, атмосфера, енергія припливів і відпливів, та ін.).

- Замінні і незамінні – ті, що можуть бути замінені (наприклад метали – пластмасами) і не можуть бути замінені іншими ресурсами (атмосферний кисень для дихання, прісна вода для пиття).

- Відтворювані і невідтворювані – ті, що принципово можна відтворити (прискорити відтворення) за рахунок застосування праці людей, і ті, що до такого відтворення не придатні (наприклад біологічний вид, екосистема та ін.).

ВИСНОВКИ

В дипломній роботі було розглянуто предметну область, проведено огляд телемедичних систем, наведена загальна інформація про телемедичні системи та їх задачі. Був досліджений метод аналізу ієрархії, область його застосування, алгоритм його роботи.

Наступним кроком був вибір програмної платформи та мови програмування. Серед всіх варіантів було обрано платформу ASP.NET, оскільки такий підхід зміг забезпечити найбільшу продуктивність роботи та швидко організувати програмну логіку роботи і зручний користувацький інтерфейс.

Необхідною умовою застосування методу аналізу ієрархії – є побудова відповідної ієрархії для потрібної предметної області. Була побудована відповідна ієрархія для застосування алгоритму в області телемедицини, яка включає більшість можливих критерії телемединчої системи для вирішення задач їх проектування та модернізації.

В дипломній роботі було розроблено комп’ютерну програму для моделювання роботи методу аналізу ієрархії в задачах проектування та модернізації телемедичних систем.

Комп’ютерну програму написано на мові програмування C#. Також було розроблено зручний інтерфейс, що містить відповідні форми та поля для полегшення роботи користувача. Програма отримує дані, введені з клавіатури, та обробляє їх згідно алгоритму методу аналізу ієрархії, знаходить значення локальних та глобальних пріоритетів, індексів збіжності даних, що демонструють коректність і точність введених даних. Результатом виконання програми є глобальні значення альтернатив, які демонструють перевагу однієї альтернативи над іншою.

Також було описано апаратні та програмні характеристики, необхідні для роботи програми. Проведено верифікацію програми з метою перевірки правильності її роботи і відповідності функціоналу та апробацію для тестування компонентів програми на відсутність помилок. Крім того було обґрунтовано

економічну доцільність розробки та впровадження цієї програми в галузь телемедицини, шляхом розрахунку норм часу на виконання алгоритму, розрахунку витрат на розробку програмного забезпечення, розрахунку ціни комп'ютерної програми і економічної ефективності від використання програмного продукту. Був знайдений термін окупності програмного продукту, що відповідає необхідним вимогам.

В розділі «Охорона праці та безпека в надзвичайних ситуаціях» було описано умови праці на робочому місці людей, що будуть працювати в області телемедицини, зокрема виявлено шкідливі і небезпечні фактори, заходи, націлені на створення умов праці, що відповідають вимогам усіх норм і стандартів з охорони праці.

АНОТАЦІЯ

Вишневський В.І. Метод аналізу ієрархії в задачах проектування та модернізації телемедичних систем.

Метою дослідження є аналіз можливих способів впровадження методу аналізу ієрархії в сферу телемедичних систем, а саме створення програмного продукту для моделювання роботи даного методу.

Предмет дослідження – метод аналізу ієрархії, що використовується для проектування та модернізації телемедичних систем.

Об’єкт дослідження – особливості впровадження методу аналізу ієрархії в телемедичні системи.

Метою дослідження є розробка комп’ютерної програми для моделювання роботи методу аналізу ієрархій в області телемедичних систем, шляхом розробки алгоритму та необхідної ієрархії. Створення графічного інтерфейсу та відповідність програмного продукту заданим вимогам.

Дана система може бути застосована в вигляді системи підтримки прийняття рішень для порівняння та вибору телемедичної системи. Вона може бути рекомендована для користування людям, що працюють в медичній сфері, телемедичним центрам.

Рік виконання дипломної роботи 2013

Рік захисту роботи 2013

Ключові слова: телемедицина, метод аналізу ієрархій, ASP.NET.

Дипломна робота містить 126 сторінок, 39 таблиць, 16 рисунків, перелік використаних джерел 20 найменувань, 1 додаток.

ANNOTATION

V.Vyshnevsky. Analytic hierarchy process in purpose of design and modernization of telemedicine systems.

The aim of the research is the analysis of possible ways of implementing the analytic hierarchy within the scope of telemedicine systems, namely the creation of software for the simulation of this method.

Research subject – analytic hierarchy process, which used for design and modernization of telemedicine systems.

Research object – the introduction of the analytic hierarchy process in telemedicine systems.

The aim of the study is developing a computer program for the simulation of the analytic hierarchy process in the field of telemedicine systems, by developing an algorithm and desired hierarchy. Creating a GUI and compliance a software product according to specified requirements.

This system can be applied as a decision support system for comparison and selection of telemedicine systems. It is recommended for people who work in the medicine, telemedicine centers.

Thesis release year 2013

Defense of the thesis year 2013

Keywords: telemedicine, the analytic hierarchy process, ASP.NET.

Thesis contains 128 pages, 39 tables, 16 figures, a list of 20 names of sources used, 1 addition.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Телемедицина [Електронний ресурс]. – Режим доступу: URL: http://www.telemedcare.ru/user_opr.php?npage=1 – Назва з екрану.
2. Телемедична система КМІС [Електронний ресурс]. – Режим доступу: URL: http://www.kmis.ru/site.nsf/pages/telemedicine.htm – Назва з екрану.
3. Телемедична система Медіалог [Електронний ресурс]. – Режим доступу: URL: http://www.medialog.ru/?tree_id=36 – Назва з екрану.
4. Стандарт HL7 [Електронний ресурс]. – Режим доступу: URL: http://uchni.com.ua/informatika/11399/index.html – Назва з екрану.
5. Саати Т. Принятие решений. Метод анализа иерархий. – М.: Радио и связь, 1993. – 278 с.
6. Саати Т. Кернс К. Аналитическое планирование. Организация систем - М.: Радио и связь, 1991. – 226 с.
7. Технології прийняття рішень [Електронний ресурс]. – Режим доступу: URL: http://citforum.ru/consulting/BI/resolution/ – Назва з екрану.
8. Платформа .NET [Електронний ресурс]. – Режим доступу: URL: http://ru.wikipedia.org/wiki/.NET – Назва з екрану.
9. Платформа ASP.NET [Електронний ресурс]. – Режим доступу: URL: http://ru.wikipedia.org/wiki/ASP.NET– Назва з екрану.
10. Екологія, порядок денний [Електронний ресурс]. – Режим доступу: URL: http://cd.greenpack.in.ua/poryadok-dennyy-na-hhi-stolittya/ – Назва з екрану.
11. Екологія, порядок денний [Електронний ресурс]. – Режим доступу: URL: http://uk.wikipedia.org/wiki/Порядок_денний_на_XXI_століття– Назва з екрану.
12. Статистика природних та екологічних факторів [Електронний ресурс]. – Режим доступу: URL: http://www.cul.com.ua/preview/Ekolog_stat-Tarasova.pdf– Назва з екрану.
13. Бейм Н. Статистические методы в биологии / Бейм Н. - М.: ИЛ., 1962. - 134 с.
14. Лебедев А.Н. Моделирование в научно-технических исследованиях / Лебедев А.Н. – М.: Радио и связь, 1989. – 224 с.

15. Організація охорони праці на виробництві [Електронний ресурс]. – Режим доступу: URL: http://www.ukrreferat.com/index.php?referat=50833&pg=1 – Назва з екрану.

16. Предмет, структура, зміст дисципліни «Охорони праці» [Електронний ресурс]. – Режим доступу: URL: http://pidruchniki.ws/1922082238244/bzhd/predmet_struktura_zmist_meta_distsiplini_osnovi_ohoroni_pratsi_zvyazok_inshimi_distsiplinami – Назва з екрану.

17. Оцінка умов праці [Електронний ресурс]. – Режим доступу: URL: http://studentbooks.com.ua/content/view/1332/76/1/9/ – Назва з екрану.

18. Витрати виробництва та їх види [Електронний ресурс]. – Режим доступу: URL: http://pidruchniki.ws/12590605/politekonomiya/vitrati_virobnitstva_valoviy_produkt_dohid_pributok - Назва з екрану.

19. ДСанПІН [Електронний ресурс]. – Режим доступу: URL: http://zakon2.rada.gov.ua/laws/show/z0293-10 - Назва з екрану.

20. Закон України «Про цивільну оборону» [Електронний ресурс]. – Режим доступу: URL: http://zakon0.rada.gov.ua/laws/show/2974-12 – Назва з екрану.

ДОДАТОК А

ЛІСТИНГ ПРОГРАМИ

Файл Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;
using DiplomQinFormVictop.Model;

namespace DiplomQinFormVictop
{
    public static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }

        /// <summary>
        /// Дістає всі  текстові значення
        /// </summary>
        /// <param name="valume"></param>
        /// <returns></returns>
        public static string GetValumeForLable(int valume = 0)
        {
            var returnValume = "";
            switch (valume)
            {
                case 1: returnValume = "Введить назву першої
альтернативи:"; break;
                case 2: returnValume = "Введить назву другої
альтернативи:"; break;
                case 3: returnValume = "Вибір телемедичної системи(Т):";
break;
                case 4: returnValume = "Моніторинг"; break;
                case 5: returnValume = "Передача даних"; break;
                case 6: returnValume = "Діагностика"; break;
                case 7: returnValume = "Збереження даних"; break;
                case 8: returnValume = "Опрацювання даних"; break;
                case 9: returnValume = "Для початку роботи введіть значення
альтернатив"; break;
                case 10: returnValume = "Обчислити"; break;
                case 11: returnValume = "Неправильно введене або не введене
значення"; break;
                case 12: returnValume = "Вага "; break;
                case 13: returnValume = "Значення вектора"; break;
```

```
                case 14: returnValume = "Коефіцієнт послідовності даних";
break;
                case 15: returnValume = "Лямда = "; break;
                case 16: returnValume = "Коефіцієнт послідовності даних(CR)
= "; break;
                case 17: returnValume = "CI ="; break;
                case 18: returnValume = "Повторити"; break;
                case 19: returnValume = "Наступний крок"; break;
                case 20: returnValume = "Інформативності"; break;
                case 21: returnValume = "Продуктивності"; break;
                case 22: returnValume = "Вартості"; break;
                case 23: returnValume = "Функціональності"; break;
                case 24: returnValume = "Надійності"; break;
                case 25: returnValume = "Масштабованості"; break;
                case 26: returnValume = "Для продовження роботи завершіть
роботу з всіма матрицями"; break;
                case 27: returnValume = "Розпочати роботу заново"; break;
                case 28: returnValume = "Діагностичні ознаки"; break;
                case 29: returnValume = "Сигнал"; break;
                case 30: returnValume = "Метод"; break;
                case 31: returnValume = "Багатозадачність"; break;
                case 32: returnValume = "Обчислювальна потужність"; break;
                case 33: returnValume = "Точність"; break;
                case 34: returnValume = "Розгортання"; break;
                case 35: returnValume = "Обслуговування"; break;
                case 36: returnValume = "Модернізація"; break;
                case 37: returnValume = "Технічна підтримка"; break;
                case 38: returnValume = "Навчання"; break;
                case 39: returnValume = "Представлення"; break;
                case 40: returnValume = "Неінвазивність"; break;
                case 41: returnValume = "Захищеність"; break;
                case 42: returnValume = "Відновлення"; break;
                case 43: returnValume = "Інтегрованість"; break;
                case 44: returnValume = "Здатність до збільшення"; break;
                case 45: returnValume = "Мобільність"; break;
                case 46: returnValume = "Глобальні пріоритети"; break;
                case 47: returnValume = "Глобальний пріорітет "; break;
                case 48: returnValume = "Результати обрахунку кращої
альтернативи"; break;
                case 49: returnValume = "Вийти"; break;
                case 50: returnValume = "Дана прогорама робroблена
Вишневським В.І.Для дипломної роботи за темою: Метод аналізу ієрархій в
задачах проектування та модернізації телемедичних систем"; break;
                default: returnValume = "undefined"; break;
            }
            return returnValume;
        }
    }
}
```

Файл DataContext.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using DiplomQinFormVictop.Model;

namespace DiplomQinFormVictop
{
    public static class DataContext
    {
        public static string Alternative1; //Вводимо назви телемедичних
систем
        public static string Alternative2; //Вводимо назви телемедичних
систем



        public static Matrix_TA TAm1 = new Matrix_TA(); // Перша матриця
5x5, що включає A1, A2, A3, A4, A5 (така буде 1)

        public static Matrix_AB ABm1 = new Matrix_AB(); // Перша матриця
6x6, що включає B1, B2, B3, B4, B5, B6 (таких буде 5)
        public static Matrix_AB ABm2 = new Matrix_AB();
        public static Matrix_AB ABm3 = new Matrix_AB();
        public static Matrix_AB ABm4 = new Matrix_AB();
        public static Matrix_AB ABm5 = new Matrix_AB();

        public static Matrix_BBX BBXm1 = new Matrix_BBX();// Перша матриця
3x3, що включає B11, B12, B13, B21..B63 (таких буде 6)
        public static Matrix_BBX BBXm2 = new Matrix_BBX();
        public static Matrix_BBX BBXm3 = new Matrix_BBX();
        public static Matrix_BBX BBXm4 = new Matrix_BBX();
        public static Matrix_BBX BBXm5 = new Matrix_BBX();
        public static Matrix_BBX BBXm6 = new Matrix_BBX();

        public static Matrix_BXC BXCm1 = new Matrix_BXC(); //Перша матриця
2x2, що включає C1, C2 (таких буде 18)
        public static Matrix_BXC BXCm2 = new Matrix_BXC();
        public static Matrix_BXC BXCm3 = new Matrix_BXC();
        public static Matrix_BXC BXCm4 = new Matrix_BXC();
        public static Matrix_BXC BXCm5 = new Matrix_BXC();
        public static Matrix_BXC BXCm6 = new Matrix_BXC();
        public static Matrix_BXC BXCm7 = new Matrix_BXC();
        public static Matrix_BXC BXCm8 = new Matrix_BXC();
        public static Matrix_BXC BXCm9 = new Matrix_BXC();
        public static Matrix_BXC BXCm10 = new Matrix_BXC();
        public static Matrix_BXC BXCm11 = new Matrix_BXC();
        public static Matrix_BXC BXCm12 = new Matrix_BXC();
        public static Matrix_BXC BXCm13 = new Matrix_BXC();
        public static Matrix_BXC BXCm14 = new Matrix_BXC();
        public static Matrix_BXC BXCm15 = new Matrix_BXC();
        public static Matrix_BXC BXCm16 = new Matrix_BXC();
        public static Matrix_BXC BXCm17 = new Matrix_BXC();
        public static Matrix_BXC BXCm18 = new Matrix_BXC();
```

```
        /// <summary>
        /// Обчислення кінцевих данних
        /// </summary>
        /// <returns></returns>
        public static LastResult CalculationGlobalPriorities()
        {
            var model = new LastResult();
            float globalB1 = TAm1.ta1 * ABm1.ab1 + TAm1.ta2 * ABm2.ab1 +
TAm1.ta3 * ABm3.ab1 + TAm1.ta4 * ABm4.ab1 + TAm1.ta5 * ABm5.ab1;
            float globalB2 = TAm1.ta1 * ABm1.ab2 + TAm1.ta2 * ABm2.ab2 +
TAm1.ta3 * ABm3.ab2 + TAm1.ta4 * ABm4.ab2 + TAm1.ta5 * ABm5.ab2;
            float globalB3 = TAm1.ta1 * ABm1.ab3 + TAm1.ta2 * ABm2.ab3 +
TAm1.ta3 * ABm3.ab3 + TAm1.ta4 * ABm4.ab3 + TAm1.ta5 * ABm5.ab3;
            float globalB4 = TAm1.ta1 * ABm1.ab4 + TAm1.ta2 * ABm2.ab4 +
TAm1.ta3 * ABm3.ab4 + TAm1.ta4 * ABm4.ab4 + TAm1.ta5 * ABm5.ab4;
            float globalB5 = TAm1.ta1 * ABm1.ab5 + TAm1.ta2 * ABm2.ab5 +
TAm1.ta3 * ABm3.ab5 + TAm1.ta4 * ABm4.ab5 + TAm1.ta5 * ABm5.ab5;
            float globalB6 = TAm1.ta1 * ABm1.ab6 + TAm1.ta2 * ABm2.ab6 +
TAm1.ta3 * ABm3.ab6 + TAm1.ta4 * ABm4.ab6 + TAm1.ta5 * ABm5.ab6;

            float globalB11 = globalB1 * BBXm1.bbx1;
            float globalB12 = globalB1 * BBXm1.bbx2;
            float globalB13 = globalB1 * BBXm1.bbx3;
            float globalB21 = globalB2 * BBXm2.bbx1;
            float globalB22 = globalB2 * BBXm2.bbx2;
            float globalB23 = globalB2 * BBXm2.bbx3;
            float globalB31 = globalB3 * BBXm3.bbx1;
            float globalB32 = globalB3 * BBXm3.bbx2;
            float globalB33 = globalB3 * BBXm3.bbx3;
            float globalB41 = globalB4 * BBXm4.bbx1;
            float globalB42 = globalB4 * BBXm4.bbx2;
            float globalB43 = globalB4 * BBXm4.bbx3;
            float globalB51 = globalB5 * BBXm5.bbx1;
            float globalB52 = globalB5 * BBXm5.bbx2;
            float globalB53 = globalB5 * BBXm5.bbx3;
            float globalB61 = globalB6 * BBXm6.bbx1;
            float globalB62 = globalB6 * BBXm6.bbx2;
            float globalB63 = globalB6 * BBXm6.bbx3;

            model.Block1 = new ResultFor2X2Matrix() { Value1 = globalB11,
Value2 = globalB12, Value3 = globalB13 };
            model.Block2 = new ResultFor2X2Matrix() { Value1 = globalB21,
Value2 = globalB22, Value3 = globalB23 };
            model.Block3 = new ResultFor2X2Matrix() { Value1 = globalB31,
Value2 = globalB32, Value3 = globalB33 };
            model.Block4 = new ResultFor2X2Matrix() { Value1 = globalB41,
Value2 = globalB42, Value3 = globalB43 };
            model.Block5 = new ResultFor2X2Matrix() { Value1 = globalB51,
Value2 = globalB52, Value3 = globalB53 };
            model.Block6 = new ResultFor2X2Matrix() { Value1 = globalB61,
Value2 = globalB62, Value3 = globalB63 };

            // Формули для обрахунку кращої альтернативи
            float globalAlternative1 = globalB11 * BXCm1.bxc1 + globalB12 *
BXCm2.bxc1 + globalB13 * BXCm3.bxc1 + globalB21 * BXCm4.bxc1 + globalB22 *
BXCm5.bxc1 +
```

```
            globalB23 * BXCm6.bxc1 + globalB31 * BXCm7.bxc1 + globalB32 *
BXCm8.bxc1 + globalB33 * BXCm9.bxc1 + globalB41 * BXCm10.bxc1 + globalB42 *
BXCm11.bxc1 +
            globalB43 * BXCm12.bxc1 + globalB51 * BXCm13.bxc1 + globalB52 *
BXCm14.bxc1 + globalB53 * BXCm15.bxc1 + globalB61 * BXCm16.bxc1 + globalB62
* BXCm17.bxc1 +
            globalB63 * BXCm18.bxc1;

            float globalAlternative2 = globalB11 * BXCm1.bxc2 + globalB12 *
BXCm2.bxc2 + globalB13 * BXCm3.bxc2 + globalB21 * BXCm4.bxc2 + globalB22 *
BXCm5.bxc2 +
            globalB23 * BXCm6.bxc2 + globalB31 * BXCm7.bxc2 + globalB32 *
BXCm8.bxc2 + globalB33 * BXCm9.bxc2 + globalB41 * BXCm10.bxc2 + globalB42 *
BXCm11.bxc2 +
            globalB43 * BXCm12.bxc2 + globalB51 * BXCm13.bxc2 + globalB52 *
BXCm14.bxc2 + globalB53 * BXCm15.bxc2 + globalB61 * BXCm16.bxc2 + globalB62
* BXCm17.bxc2 +
            globalB63 * BXCm18.bxc2;

            model.GlobalAlternative1 = globalAlternative1;
            model.GlobalAlternative2 = globalAlternative2;

            return model;
        }
    }

    /// <summary>
    /// Модель кінцевих даних
    /// </summary>
    public class LastResult
    {
        public ResultFor2X2Matrix Block1 = new ResultFor2X2Matrix();
        public ResultFor2X2Matrix Block2 = new ResultFor2X2Matrix();
        public ResultFor2X2Matrix Block3 = new ResultFor2X2Matrix();
        public ResultFor2X2Matrix Block4 = new ResultFor2X2Matrix();
        public ResultFor2X2Matrix Block5 = new ResultFor2X2Matrix();
        public ResultFor2X2Matrix Block6 = new ResultFor2X2Matrix();

        public float GlobalAlternative1, GlobalAlternative2;

    }
    public class ResultFor2X2Matrix
    {
        public float Value1, Value2, Value3;
    }
}
```

Файл Matrix_AB.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace DiplomQinFormVictop.Model
{
    public class Matrix_AB
    {
        public float ab1 = 0; // Інформативності
        public float ab2 = 0; // Продуктивності
        public float ab3 = 0; // Вартості
        public float ab4 = 0; // Функціональності
        public float ab5 = 0; // Надійності
        public float ab6 = 0; // Масштабованості
        public float CR = 0;  // - коефіцієнт послідовності даних
        public float Lambda = 0;
        public float CI = 0;

        float[,] basic_matrix = new float[6, 6];

        // метод обробки матриці 6х6
        public void Handle()
        {
            // шукаємо добуток
            float a0 = basic_matrix[0, 0] * basic_matrix[0, 1] *
basic_matrix[0, 2] * basic_matrix[0, 3] * basic_matrix[0, 4] *
basic_matrix[0, 5];
            float a1 = basic_matrix[1, 0] * basic_matrix[1, 1] *
basic_matrix[1, 2] * basic_matrix[1, 3] * basic_matrix[1, 4] *
basic_matrix[1, 5];
            float a2 = basic_matrix[2, 0] * basic_matrix[2, 1] *
basic_matrix[2, 2] * basic_matrix[2, 3] * basic_matrix[2, 4] *
basic_matrix[2, 5];
            float a3 = basic_matrix[3, 0] * basic_matrix[3, 1] *
basic_matrix[3, 2] * basic_matrix[3, 3] * basic_matrix[3, 4] *
basic_matrix[3, 5];
            float a4 = basic_matrix[4, 0] * basic_matrix[4, 1] *
basic_matrix[4, 2] * basic_matrix[4, 3] * basic_matrix[4, 4] *
basic_matrix[4, 5];
            float a5 = basic_matrix[5, 0] * basic_matrix[5, 1] *
basic_matrix[5, 2] * basic_matrix[5, 3] * basic_matrix[5, 4] *
basic_matrix[5, 5];

            a0 = (float)Math.Pow(a0, 0.16666);
            a1 = (float)Math.Pow(a1, 0.16666);
            a2 = (float)Math.Pow(a2, 0.16666);
            a3 = (float)Math.Pow(a3, 0.16666);
            a4 = (float)Math.Pow(a4, 0.16666);
            a5 = (float)Math.Pow(a5, 0.16666);

            float sumA = a0 + a1 + a2 + a3 + a4 + a5;
```

```
            //Знаходимо вектор
            ab1 = a0 / sumA;
            ab2 = a1 / sumA;
            ab3 = a2 / sumA;
            ab4 = a3 / sumA;
            ab5 = a4 / sumA;
            ab6 = a5 / sumA;
        }

        // тут знаходимо коефіцієнт послідовності даних
        public void Convergence()
        {
            Lambda = ab1 * (basic_matrix[0, 0] + basic_matrix[1, 0] +
basic_matrix[2, 0] + basic_matrix[3, 0] + basic_matrix[4, 0] +
basic_matrix[5, 0]) +
            ab2 * (basic_matrix[0, 1] + basic_matrix[1, 1] +
basic_matrix[2, 1] + basic_matrix[3, 1] + basic_matrix[4, 1] +
basic_matrix[5, 1]) +
            ab3 * (basic_matrix[0, 2] + basic_matrix[1, 2] +
basic_matrix[2, 2] + basic_matrix[3, 2] + basic_matrix[4, 2] +
basic_matrix[5, 2]) +
            ab4 * (basic_matrix[0, 3] + basic_matrix[1, 3] +
basic_matrix[2, 3] + basic_matrix[3, 3] + basic_matrix[4, 3] +
basic_matrix[5, 3]) +
            ab5 * (basic_matrix[0, 4] + basic_matrix[1, 4] +
basic_matrix[2, 4] + basic_matrix[3, 4] + basic_matrix[4, 4] +
basic_matrix[5, 4]) +
            ab6 * (basic_matrix[0, 5] + basic_matrix[1, 5] +
basic_matrix[2, 5] + basic_matrix[3, 5] + basic_matrix[4, 5] +
basic_matrix[5, 5]);
            CI = (Lambda - 6) / 5;
            CR = CI / 1.24F;
        }

        // встановлює значення в матрицю
        public void Enter(float[] value)
        {
            for (int i = 0; i < 6; i++)
                for (int j = 0; j < 6; j++)
                {
                    var element = (i * 6) + j;
                    basic_matrix[i, j] = value[element];
                }
        }
    }
}
```

Файл Matrix_BBX.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace DiplomQinFormVictop.Model
{
    public class Matrix_BBX
    {
        public float bbx1 = 0;
        public float bbx2 = 0;
        public float bbx3 = 0;
        public float CR = 0;  // - коефіцієнт послідовності даних
        public float Lambda = 0;
        public float CI = 0;

        float[,] basic_matrix = new float[3, 3];

        // метод обробки матриці 3x3
        public void Handle()
        {
            // шукаємо добуток
            float a0 = basic_matrix[0, 0] * basic_matrix[0, 1] *
basic_matrix[0, 2];
            float a1 = basic_matrix[1, 0] * basic_matrix[1, 1] *
basic_matrix[1, 2];
            float a2 = basic_matrix[2, 0] * basic_matrix[2, 1] *
basic_matrix[2, 2];

            // знаходимо вектор У (він являє собою ваги дуг ієрархії)
            a0 = (float)Math.Pow(a0, 0.3333);
            a1 = (float)Math.Pow(a1, 0.3333);
            a2 = (float)Math.Pow(a2, 0.3333);

            float sumA = a0 + a1 + a2;

            //Знаходимо вектор
            bbx1 = a0 / sumA;
            bbx2 = a1 / sumA;
            bbx3 = a2 / sumA;
        }

        // тут знаходимо коефіцієнт послідовності даних
        public void Convergence()
        {
            Lambda = bbx1 * (basic_matrix[0, 0] + basic_matrix[1, 0] +
basic_matrix[2, 0]) +
            bbx2 * (basic_matrix[0, 1] + basic_matrix[1, 1] +
basic_matrix[2, 1]) +
            bbx3 * (basic_matrix[0, 2] + basic_matrix[1, 2] +
basic_matrix[2, 2]);

            CI = (Lambda - 3) / 2;
            CR = CI / 0.58F;
```

```
        }

        // встановлює значення в матрицю
        public void Enter(float[] value)
        {
            for (int i = 0; i < 3; i++)
                for (int j = 0; j < 3; j++)
                {
                    var element = (i * 3) + j;
                    basic_matrix[i, j] = value[element];
                }
        }
    }
}
```

Файл Matrix_BBX.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace DiplomQinFormVictop.Model
{
    public class Matrix_BXC
    {
        public float bxc1 = 0;
        public float bxc2 = 0;

        float[,] basic_matrix = new float[2, 2];

        // метод обробки матриці 2x2
        public void Handle()
        {
            // шукаємо добуток
            float a0 = basic_matrix[0, 0] * basic_matrix[0, 1];
            float a1 = basic_matrix[1, 0] * basic_matrix[1, 1];

            // знаходимо вектор У (він являє собою ваги дуг ієрархії)
            a0 = (float)Math.Pow(a0, 0.5);
            a1 = (float)Math.Pow(a1, 0.5);

            float sumA = a0 + a1;

            //Знаходимо вектор
            bxc1 = a0 / sumA;
            bxc2 = a1 / sumA;

        }

        // встановлює значення в матрицю
        public void Enter(float[] value)
        {
            for (int i = 0; i < 2; i++)
                for (int j = 0; j < 2; j++)
                {
                    var element = (i * 2) + j;
                    basic_matrix[i, j] = value[element];
                }
        }
    }
}
```

Файл Matrix_TA.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace DiplomQinFormVictop.Model
{
    public class Matrix_TA
    {
        public float ta1 = 0; // - вага Моніторингу
        public float ta2 = 0; // - вага Передачі даних
        public float ta3 = 0; // - вага Діагностики
        public float ta4 = 0; // - вага Збереження даних
        public float ta5 = 0; // - вага Опрацювання даних
        public float CR = 0;  // - коефіцієнт послідовності даних
        public float Lambda = 0;
        public float CI = 0;

        float[,] basic_matrix = new float[5, 5];

        // метод обробки матриці 5x5
        public void Handle()
        {

            // шукаємо добуток
            float a0 = basic_matrix[0, 0] * basic_matrix[0, 1] *
basic_matrix[0, 2] * basic_matrix[0, 3] * basic_matrix[0, 4];
            float a1 = basic_matrix[1, 0] * basic_matrix[1, 1] *
basic_matrix[1, 2] * basic_matrix[1, 3] * basic_matrix[1, 4];
            float a2 = basic_matrix[2, 0] * basic_matrix[2, 1] *
basic_matrix[2, 2] * basic_matrix[2, 3] * basic_matrix[2, 4];
            float a3 = basic_matrix[3, 0] * basic_matrix[3, 1] *
basic_matrix[3, 2] * basic_matrix[3, 3] * basic_matrix[3, 4];
            float a4 = basic_matrix[4, 0] * basic_matrix[4, 1] *
basic_matrix[4, 2] * basic_matrix[4, 3] * basic_matrix[4, 4];

            a0 = (float)Math.Pow(a0, 0.2);
            a1 = (float)Math.Pow(a1, 0.2);
            a2 = (float)Math.Pow(a2, 0.2);
            a3 = (float)Math.Pow(a3, 0.2);
            a4 = (float)Math.Pow(a4, 0.2);

            float sumA = a0 + a1 + a2 + a3 + a4;

            //Знаходимо вектор (ваги Моніторингу, Передачі даних,
Діагностики, Збереження даних, Опрацювання даних)
            ta1 = a0 / sumA;
            ta2 = a1 / sumA;
            ta3 = a2 / sumA;
            ta4 = a3 / sumA;
            ta5 = a4 / sumA;
        }

        // тут знаходимо коефіцієнт послідовності даних
```

```
        public void Convergence(ref float CR)
        {
            float lambda = ta1 * (basic_matrix[0, 0] + basic_matrix[1, 0] +
basic_matrix[2, 0] + basic_matrix[3, 0] + basic_matrix[4, 0]) +
            ta2 * (basic_matrix[0, 1] + basic_matrix[1, 1] +
basic_matrix[2, 1] + basic_matrix[3, 1] + basic_matrix[4, 1]) +
            ta3 * (basic_matrix[0, 2] + basic_matrix[1, 2] +
basic_matrix[2, 2] + basic_matrix[3, 2] + basic_matrix[4, 2]) +
            ta4 * (basic_matrix[0, 3] + basic_matrix[1, 3] +
basic_matrix[2, 3] + basic_matrix[3, 3] + basic_matrix[4, 3]) +
            ta5 * (basic_matrix[0, 4] + basic_matrix[1, 4] +
basic_matrix[2, 4] + basic_matrix[3, 4] + basic_matrix[4, 4]);
            Lambda = lambda;
            CI = (lambda - 5) / 4;
            CR = CI / 1.12F;
        }

        // встановлює значення в матрицю
        public void Enter(float[] value)
        {
            for (int i = 0; i < 5; i++)
                for (int j = 0; j < 5; j++)
                {
                    var element = (i*5) + j;
                    basic_matrix[i, j] = value[element];
                }
        }
    }
}
```

Файл Form1.Designer.cs

```
namespace DiplomQinFormVictop
{
    partial class Form1
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be
disposed; otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.TextBox textBox1;
        private System.Windows.Forms.TextBox textBox2;
        private System.Windows.Forms.TextBox textBox3;
        private System.Windows.Forms.TextBox textBox4;
        private System.Windows.Forms.TextBox textBox5;
        private System.Windows.Forms.TextBox textBox6;
        private System.Windows.Forms.TextBox textBox7;
        private System.Windows.Forms.TextBox textBox8;
        private System.Windows.Forms.TextBox textBox9;
        private System.Windows.Forms.TextBox textBox10;
        private System.Windows.Forms.TextBox textBox11;
        private System.Windows.Forms.TextBox textBox12;
        private System.Windows.Forms.TextBox textBox13;
        private System.Windows.Forms.TextBox textBox14;
        private System.Windows.Forms.TextBox textBox15;
        private System.Windows.Forms.TextBox textBox16;
        private System.Windows.Forms.TextBox textBox17;
        private System.Windows.Forms.TextBox textBox18;
        private System.Windows.Forms.TextBox textBox19;
        private System.Windows.Forms.TextBox textBox20;
        private System.Windows.Forms.TextBox textBox21;
        private System.Windows.Forms.TextBox textBox22;
        private System.Windows.Forms.TextBox textBox23;
        private System.Windows.Forms.TextBox textBox24;
        private System.Windows.Forms.TextBox textBox25;
        private System.Windows.Forms.TextBox textBox26;
        private System.Windows.Forms.TextBox textBox27;
        private System.Windows.Forms.Label label3;
```

```
        private System.Windows.Forms.Label label4;
        private System.Windows.Forms.Label label5;
        private System.Windows.Forms.Label label6;
        private System.Windows.Forms.Label label7;
        private System.Windows.Forms.Label label8;
        private System.Windows.Forms.Label label9;
        private System.Windows.Forms.Label label10;
        private System.Windows.Forms.Label label11;
        private System.Windows.Forms.Label label12;
        private System.Windows.Forms.Label label13;
        private System.Windows.Forms.Label label14;
        private System.Windows.Forms.Button button1;
        private System.Windows.Forms.TextBox textBox28;
        private System.Windows.Forms.GroupBox groupBox1;
        private System.Windows.Forms.GroupBox groupBox2;
        private System.Windows.Forms.Label label19;
        private System.Windows.Forms.Label label18;
        private System.Windows.Forms.Label label17;
        private System.Windows.Forms.Label label16;
        private System.Windows.Forms.Label label15;
        private System.Windows.Forms.Label label22;
        private System.Windows.Forms.Label label21;
        private System.Windows.Forms.Label label20;
        private System.Windows.Forms.Button button2;
        private System.Windows.Forms.Button button3;

    }
}
```

Файл Form2.Designer.cs

```
namespace DiplomQinFormVictop
{
    partial class Form2
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be
disposed; otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        private MyControl.WorckMatrix6X6 worckMatrix6X61;
        private MyControl.WorckMatrix6X6 worckMatrix6X62;
        private MyControl.WorckMatrix6X6 worckMatrix6X63;
        private MyControl.WorckMatrix6X6 worckMatrix6X64;
        private System.Windows.Forms.Splitter splitter1;
        private MyControl.WorckMatrix6X6 worckMatrix6X65;
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.Button button1;
        private System.Windows.Forms.Button button2;
        private System.Windows.Forms.GroupBox groupBox1;
    }
}
```

Файл Form3.Designer.cs

```
namespace DiplomQinFormVictop
{
    partial class Form3
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be
disposed; otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code


        private MyControl.Work3X3Matrix work3X3Matrix1;
        private MyControl.Work3X3Matrix work3X3Matrix2;
        private MyControl.Work3X3Matrix work3X3Matrix3;
        private MyControl.Work3X3Matrix work3X3Matrix4;
        private MyControl.Work3X3Matrix work3X3Matrix5;
        private MyControl.Work3X3Matrix work3X3Matrix6;
        private System.Windows.Forms.TextBox textBox1;
        private System.Windows.Forms.Button button1;
        private System.Windows.Forms.Button button2;

    }
}
```

Файл Form4.Designer.cs

```
namespace DiplomQinFormVictop
{
    partial class Form4
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be
disposed; otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        private MyControl.Work2X2Matrix work2X2Matrix1;
        private MyControl.Work2X2Matrix work2X2Matrix2;
        private MyControl.Work2X2Matrix work2X2Matrix3;
        private MyControl.Work2X2Matrix work2X2Matrix4;
        private MyControl.Work2X2Matrix work2X2Matrix5;
        private MyControl.Work2X2Matrix work2X2Matrix6;
        private MyControl.Work2X2Matrix work2X2Matrix7;
        private MyControl.Work2X2Matrix work2X2Matrix8;
        private MyControl.Work2X2Matrix work2X2Matrix9;
        private MyControl.Work2X2Matrix work2X2Matrix10;
        private MyControl.Work2X2Matrix work2X2Matrix11;
        private MyControl.Work2X2Matrix work2X2Matrix12;
        private MyControl.Work2X2Matrix work2X2Matrix13;
        private MyControl.Work2X2Matrix work2X2Matrix14;
        private MyControl.Work2X2Matrix work2X2Matrix15;
        private MyControl.Work2X2Matrix work2X2Matrix16;
        private MyControl.Work2X2Matrix work2X2Matrix17;
        private MyControl.Work2X2Matrix work2X2Matrix18;
        private System.Windows.Forms.TextBox textBox1;
        private System.Windows.Forms.Button button2;
        private System.Windows.Forms.Button button1;
        private System.Windows.Forms.GroupBox groupBox1;
    }
}
```

Файл Form5.Designer.cs

```
namespace DiplomQinFormVictop
{
    partial class Form5
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be
disposed; otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.groupBox1 = new System.Windows.Forms.GroupBox();
            this.groupBox2 = new System.Windows.Forms.GroupBox();
            this.label2 = new System.Windows.Forms.Label();
            this.label1 = new System.Windows.Forms.Label();
            this.button1 = new System.Windows.Forms.Button();
            this.button2 = new System.Windows.Forms.Button();
            this.textBox1 = new System.Windows.Forms.TextBox();
            this.showGlobalResilt6 = new
DiplomQinFormVictop.MyControl.ShowGlobalResilt();
            this.showGlobalResilt5 = new
DiplomQinFormVictop.MyControl.ShowGlobalResilt();
            this.showGlobalResilt4 = new
DiplomQinFormVictop.MyControl.ShowGlobalResilt();
            this.showGlobalResilt3 = new
DiplomQinFormVictop.MyControl.ShowGlobalResilt();
            this.showGlobalResilt2 = new
DiplomQinFormVictop.MyControl.ShowGlobalResilt();
            this.showGlobalResilt1 = new
DiplomQinFormVictop.MyControl.ShowGlobalResilt();
            this.groupBox1.SuspendLayout();
            this.groupBox2.SuspendLayout();
            this.SuspendLayout();
            //
            // groupBox1
```

```
            //
            this.groupBox1.Controls.Add(this.showGlobalResilt6);
            this.groupBox1.Controls.Add(this.showGlobalResilt5);
            this.groupBox1.Controls.Add(this.showGlobalResilt4);
            this.groupBox1.Controls.Add(this.showGlobalResilt3);
            this.groupBox1.Controls.Add(this.showGlobalResilt2);
            this.groupBox1.Controls.Add(this.showGlobalResilt1);
            this.groupBox1.Font = new System.Drawing.Font("Microsoft Sans
Serif", 12F, System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.groupBox1.Location = new System.Drawing.Point(12, 12);
            this.groupBox1.Name = "groupBox1";
            this.groupBox1.Size = new System.Drawing.Size(513, 481);
            this.groupBox1.TabIndex = 0;
            this.groupBox1.TabStop = false;
            this.groupBox1.Text = "Value46";
            //
            // groupBox2
            //
            this.groupBox2.Controls.Add(this.label2);
            this.groupBox2.Controls.Add(this.label1);
            this.groupBox2.Font = new System.Drawing.Font("Microsoft Sans
Serif", 12F, System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.groupBox2.Location = new System.Drawing.Point(531, 12);
            this.groupBox2.Name = "groupBox2";
            this.groupBox2.Size = new System.Drawing.Size(490, 158);
            this.groupBox2.TabIndex = 1;
            this.groupBox2.TabStop = false;
            this.groupBox2.Text = "Value48";
            //
            // label2
            //
            this.label2.AutoSize = true;
            this.label2.Location = new System.Drawing.Point(16, 88);
            this.label2.Name = "label2";
            this.label2.Size = new System.Drawing.Size(51, 20);
            this.label2.TabIndex = 1;
            this.label2.Text = "label2";
            //
            // label1
            //
            this.label1.AutoSize = true;
            this.label1.Location = new System.Drawing.Point(16, 45);
            this.label1.Name = "label1";
            this.label1.Size = new System.Drawing.Size(51, 20);
            this.label1.TabIndex = 0;
            this.label1.Text = "label1";
            //
            // button1
            //
            this.button1.Font = new System.Drawing.Font("Microsoft Sans
Serif", 12F, System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.button1.Location = new System.Drawing.Point(531, 461);
            this.button1.Name = "button1";
            this.button1.Size = new System.Drawing.Size(98, 31);
            this.button1.TabIndex = 2;
```

```
            this.button1.Text = "Value49";
            this.button1.UseVisualStyleBackColor = true;
            this.button1.Click += new
System.EventHandler(this.button1_Click);
            //
            // button2
            //
            this.button2.Font = new System.Drawing.Font("Microsoft Sans
Serif", 12F, System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.button2.Location = new System.Drawing.Point(635, 461);
            this.button2.Name = "button2";
            this.button2.Size = new System.Drawing.Size(241, 31);
            this.button2.TabIndex = 3;
            this.button2.Text = "Value27";
            this.button2.UseVisualStyleBackColor = true;
            this.button2.Click += new
System.EventHandler(this.button2_Click);
            //
            // textBox1
            //
            this.textBox1.Font = new System.Drawing.Font("Microsoft Sans
Serif", 12F, System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.textBox1.Location = new System.Drawing.Point(531, 177);
            this.textBox1.Multiline = true;
            this.textBox1.Name = "textBox1";
            this.textBox1.ReadOnly = true;
            this.textBox1.Size = new System.Drawing.Size(490, 278);
            this.textBox1.TabIndex = 4;
            this.textBox1.Text = "Value50";
            //
            // showGlobalResilt6
            //
            this.showGlobalResilt6.Font = new
System.Drawing.Font("Microsoft Sans Serif", 8.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((byte)(204)));
            this.showGlobalResilt6.Location = new System.Drawing.Point(7,
417);
            this.showGlobalResilt6.Margin = new
System.Windows.Forms.Padding(4, 5, 4, 5);
            this.showGlobalResilt6.Name = "showGlobalResilt6";
            this.showGlobalResilt6.Size = new System.Drawing.Size(478, 68);
            this.showGlobalResilt6.TabIndex = 5;
            //
            // showGlobalResilt5
            //
            this.showGlobalResilt5.Font = new
System.Drawing.Font("Microsoft Sans Serif", 8.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((byte)(204)));
            this.showGlobalResilt5.Location = new System.Drawing.Point(7,
339);
            this.showGlobalResilt5.Margin = new
System.Windows.Forms.Padding(4, 5, 4, 5);
            this.showGlobalResilt5.Name = "showGlobalResilt5";
            this.showGlobalResilt5.Size = new System.Drawing.Size(478, 68);
```

```
            this.showGlobalResilt5.TabIndex = 4;
            //
            // showGlobalResilt4
            //
            this.showGlobalResilt4.Font = new
System.Drawing.Font("Microsoft Sans Serif", 8.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((byte)(204)));
            this.showGlobalResilt4.Location = new System.Drawing.Point(7,
261);
            this.showGlobalResilt4.Margin = new
System.Windows.Forms.Padding(4, 5, 4, 5);
            this.showGlobalResilt4.Name = "showGlobalResilt4";
            this.showGlobalResilt4.Size = new System.Drawing.Size(478, 68);
            this.showGlobalResilt4.TabIndex = 3;
            //
            // showGlobalResilt3
            //
            this.showGlobalResilt3.Font = new
System.Drawing.Font("Microsoft Sans Serif", 8.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((byte)(204)));
            this.showGlobalResilt3.Location = new System.Drawing.Point(7,
183);
            this.showGlobalResilt3.Margin = new
System.Windows.Forms.Padding(4, 5, 4, 5);
            this.showGlobalResilt3.Name = "showGlobalResilt3";
            this.showGlobalResilt3.Size = new System.Drawing.Size(478, 68);
            this.showGlobalResilt3.TabIndex = 2;
            //
            // showGlobalResilt2
            //
            this.showGlobalResilt2.Font = new
System.Drawing.Font("Microsoft Sans Serif", 8.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((byte)(204)));
            this.showGlobalResilt2.Location = new System.Drawing.Point(7,
105);
            this.showGlobalResilt2.Margin = new
System.Windows.Forms.Padding(4, 5, 4, 5);
            this.showGlobalResilt2.Name = "showGlobalResilt2";
            this.showGlobalResilt2.Size = new System.Drawing.Size(478, 68);
            this.showGlobalResilt2.TabIndex = 1;
            //
            // showGlobalResilt1
            //
            this.showGlobalResilt1.Font = new
System.Drawing.Font("Microsoft Sans Serif", 8.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((byte)(204)));
            this.showGlobalResilt1.Location = new System.Drawing.Point(7,
27);
            this.showGlobalResilt1.Margin = new
System.Windows.Forms.Padding(4, 5, 4, 5);
            this.showGlobalResilt1.Name = "showGlobalResilt1";
            this.showGlobalResilt1.Size = new System.Drawing.Size(478, 68);
            this.showGlobalResilt1.TabIndex = 0;
            //
```

```
            // Form5
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(1049, 504);
            this.Controls.Add(this.textBox1);
            this.Controls.Add(this.button2);
            this.Controls.Add(this.button1);
            this.Controls.Add(this.groupBox2);
            this.Controls.Add(this.groupBox1);
            this.Name = "Form5";
            this.StartPosition =
System.Windows.Forms.FormStartPosition.CenterScreen;
            this.Text = "Form5";
            this.FormClosing += new
System.Windows.Forms.FormClosingEventHandler(this.Form5_FormClosing);
            this.Load += new System.EventHandler(this.Form5_Load);
            this.groupBox1.ResumeLayout(false);
            this.groupBox2.ResumeLayout(false);
            this.groupBox2.PerformLayout();
            this.ResumeLayout(false);
            this.PerformLayout();

        }

        #endregion

        private System.Windows.Forms.GroupBox groupBox1;
        private MyControl.ShowGlobalResilt showGlobalResilt6;
        private MyControl.ShowGlobalResilt showGlobalResilt5;
        private MyControl.ShowGlobalResilt showGlobalResilt4;
        private MyControl.ShowGlobalResilt showGlobalResilt3;
        private MyControl.ShowGlobalResilt showGlobalResilt2;
        private MyControl.ShowGlobalResilt showGlobalResilt1;
        private System.Windows.Forms.GroupBox groupBox2;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.Button button1;
        private System.Windows.Forms.Button button2;
        private System.Windows.Forms.TextBox textBox1;
    }
}
```