Міністерство освіти і науки України
Центральноукраїнський національний технічний університет

Т Е Х Н О Л О Г І Ї П Р О Е К Т У В А Н Н Я К О М П ’ Ю Т Е Р Н И Х С И С Т Е М

НАВЧАЛЬНИЙ ПОСІБНИК

Кропивницький
Видавець Лисенко В.Ф.
2017

ББК 30.2-5-05
Т38
УДК 004

Рекомендовано вченою радою КНТУ як навчальний посібник для студентів вищих навчальних закладів
Протокол № 9 від 06.06.2016 р.

Рецензенти:

О.А. Смірнов д.т.н., проф., завідувач кафедри програмування та захисту інформації Центральноукраїнського національного технічного університету;

С. Д. Паращук к.ф-м.н., доцент, в.о. завідувача кафедри інформатики Центральноукраїнського державного педагогічного університету.

Савеленко О.К., Якименко Н.М., Колодочкіна А.В., Сорокін В.В.
Т38 Технології проектування комп'ютерних систем: Навчальний посібник. - Кропивницький: Лисенко В.Ф., 2017. - 308 с.

У навчальному посібнику, складеному за навчальною програмою курсу «Технології проектування комп'ютерних систем», основну увагу приділено розкриттю тем курсу та вирішенню практичних завдань.

Призначений для студентів, що навчаються за спеціальністю: «Комп'ютерна інженерія», а також може буде корисним спеціалістам з автоматизації систем проектування друкованих плат.

ЗМІСТ

ТЕХНОЛОГІЇ ПРОЕКТУВАННЯ КОМП'ЮТЕРНИХ СИСТЕМ ТА МЕРЕЖ. СУТНІСТЬ ДИСЦИПЛІНИ, ОБЛАСТЬ ЇЇ ЗАСТОСУВАННЯ

1. Вступ
2. Загальні положення і задачі створення САПР
3. Вимоги, що пред'являються до САПР
4. Реалізація структури САПР

1 Вступ

Комп'ютеризація - один з найважливіших важелів науково-технічного прогресу. Адже кількість знову розроблювальних приладобудівними галузями промисловості виробів подвоюється кожні 15 років, а їхня складність - кожні 10 років (в окремих галузях техніки ці показники ще вищі), вимоги до термінів і якості розробок безупинно ростуть. До останнього часу виникаючі проблеми вирішувалися в основному за рахунок постійно збільшуваної кількості інженерно-технічного персоналу (ІТП) і, частково, за рахунок рішення задач підвищення продуктивності праці проектувальників. Такий екстенсивний шлях розвитку продуктивності неефективний, про це говорять наступні факти: у світі продуктивності праці за останні 100 років у виробництві зросла в середньому на 100% , а в проектуванні - на 20% .

Впровадження засобів обчислювальної техніки (ЗОТ) у практику проектування на системній основі, створення систем

автоматизованого проектування дозволяють усунути ці протиріччя.

Застосування математичних методів і ЗОТ на всіх етапах створення і серійного випуску виробів електронної техніки (ВЕТ) і радіоелектронної апаратури (РЕА) дає значний економічний ефект, найбільша ефективність застосування ЗОТ досягається при системному підході до розв'язуваної проблеми.

Можна виділити наступні автоматизовані системи, що беруть участь у загальному циклі створення нового виробу й організації його серійного випуску на підприємствах [18]:

- АСНД (автоматизована система наукових досліджень);

- САПР (система автоматизованого проектування);

- АСКТП(автоматизована система керуванням технологічними процесами);

- АСТПВ (автоматизована система технологічної підготовки виробництва);

- АСУП (автоматизована система керуванням виробництва на рівні підприємства);

- АСУ (автоматизована система керуванням виробництва на рівні галузі).

Для максимальної ефективності всі автоматизовані системи, які експлуатуються в єдиному комплексі, повинні бути взаємозалежні і погоджені по вхідній і вихідній інформації, потужності обчислювальних засобів, організації обробки даних.

Під автоматизацією проектування розуміють застосування ПК у процесі проектування технічних об'єктів, це один з головних напрямків технічного прогресу. Це пояснюється тим, що промисловий потенціал країни визначається не тільки наявністю і можливостями виготовлення нової техніки, але і можливостями її проектування у стислі терміни. І якщо конвеєри для масового виробництва виробів в наявності у всіх галузях промисловості, то час створення конвеєрів для масового проектування нових виробів тільки настає. На конвеєр повинні бути поставлені розумові рухи висококваліфікованого інженера-проектувальника. САПР, власне кажучи, є конвеєром для проектування відповідного виробу.

Історія розвитку САПР коротка [21]. Мабуть, важко назвати іншу область людської діяльності, що розвивалася б з такою швидкістю. В історії САПР можна умовно виділити 3 періоди:

1) 1950-1960 р. – технічні дослідження можливості вирішення електротехнічних і конструкторських задач на ЕОМ і створення перших програм для рішення цих задач;

2) 1960-1970 р. – розробка методів, алгоритмів і програм вирішення окремих задач з різних етапів проектування (складання математичних моделей електронних схем, аналіз статичного і динамічного режиму їхньої роботи, параметрична оптимізація, статистичний аналіз і ін.);

3) з 1970р. – розробка САПР, продовження робіт, що характеризують перші два періоди.

2 Загальні положення і задачі створення САПР

Технічні вимоги, які пред'являються до складного радіоелектронного устаткування, безупинно підвищуються, особливо до надійності, габаритних розмірів, маси і вартості. Необхідність при постійно зростаючій складності електронних пристроїв, утримувати ці пристрої в межах розумних фізичних розмірів, призвела до ідеї мініатюризації електронних компонентів. Відповідно, виникла необхідність переходу від простих форм монтажу на шасі до компонентів на друкованих платах, від схем на друкованих платах до модульних конструкцій, від модульних конструкцій до напівпровідникових і гібридних мікросхем, і навіть взятий напрямок на їх мікромініатюрне виконання [6].

Розробка САПР спрямована на рішення задач:

- мінімізації тривалості проектування;

- забезпечення проектування виробів високої складності ($10^6 \ldots 10^7$ елементів на систему);

- підвищення якості проектування (головним чином – за рахунок безпомилковості).

Скорочення тривалості проектування обумовлюється можливістю вирішення найбільш трудомістких задач з використанням ЗОТ, бібліотек компонентів, стандартизації програм і т.д.

Ефективність скорочення тривалості проектування виражається в одержанні економічного ефекту від упровадження РЕА раніше наміченого терміну.

Проектування виробів високої складності без застосування ЕОМ практично неможливе через: тривалість терміну проектування; складність виключення помилок.

При проектуванні без застосування ЕОМ , наприклад ІС, точність розрахунку електричних параметрів невелика і приходиться робити 3-4 технологічних прогони по технологічному процесу виготовлення з наступним корегуванням електричної схеми. Число прогонів визначається кількістю помилок, допущених проектувальником, тобто в силу вступає так називаний людський фактор. Проектування за допомогою ЕОМ дозволяє узагальнити інтелект багатьох проектувальників і практично виключити помилки при проведенні процесу проектування. Машинні методи проектування розвивалися поетапно:

- на початку створювалися окремі програми для рішення різних задач проектування в режимі пакетної обробки на універсальних ЕОМ;

- зі збільшенням складності проектуємих виробів багаторазові виходи на ЕОМ, з метою налагодження нових програм, тривалість їхнього налагодження із уже наявними, призвели до негативних результатів: частка людського фактора з його помилками була по колишньому висока і значно подовжувала часовий цикл проектування;

- виникає необхідність не стільки автоматизації рішення задач на окремих етапах проектування, скільки створення на базі потужних ЕОМ високопродуктивних систем проектування ВЕТ і РЕА. За допомогою цих систем потрібно було вирішувати задачі всіх етапів проектування: від одержання ТЗ до одержання інформації на машинних носіях інформації (МНІ) для їхнього виготовлення на програмно управляємому устаткуванні .

Для рішення цих задач потрібен був уже системний підхід, тому що САПР ВЕТ і РЕА взаємозалежні.

Проблеми САПР і їхня побудова аналогічні проблемам побудови будь-яких складних систем. Містять у собі наступні аспекти:

- технологічні;
- концептуальні;
- методологічні;
- теоретичні.

Технологічні аспекти стосуються побудови системного програмного комплексу САПР, тобто:

- структури баз даних;
- системи управління БД;
- операційних систем;
- управляючих програм;
- мов проектування і програмування засобів автоматизації графічних робіт;
- засобів підготовки документації.

Концептуальні аспекти стосуються побудови системи принципів автоматизованого проектування, тобто САПР- це система для створення:

- об’єктів проектування засобами автоматизації обчислень;

- системи одержання інформації й автоматизації її обробки;

- організацій цільового людино-машинного процесу проектування;

- рішення задач управління процесом проектування.

Концептуальна структура САПР включає наступні складові:

- бібліотеку моделей об’єктів і процесів проектування;

- бібліотеку процедур, що забезпечують побудову проектних рішень;

- систему інформації;

- інструментальні засоби побудови баз даних;

- технологію створення прикладних, системних і сервісних програм;

- програмні і технічні засоби відновлення і перетворення систем.

Методологічні аспекти припускають побудову системи наукових поглядів і структуру відносин розробників і користувачів САПР. Дані аспекти визначають вибір того або іншого маршруту проектування.

Теоретичні аспекти – це формування задач теорії автоматизованого проектування і створення апарата такої

теорії. Тобто, це аналіз можливості рішення визначеної задачі проектування, аналіз загального проектного рішення, побудова законів функціонування САПР і її підсистем на всіх етапах проектування.

Таким чином, основні задачі по створенню САПР полягають у наступному:

1) Розробка методики проектування ВЕТ і РЕА, тобто визначення:

- етапів проектування;

- математичне формулювання задач кожного етапу;

- вибір чисельних методів рішення задач;

- аналіз можливості рішення цих задач за допомогою ЕОМ;

2) Розробка математичного забезпечення;

3) Створення системи технічного забезпечення: ЕОМ, периферійні пристрої, редагування і документування алфавітно-цифрової і графічної інформації з ієрархічного або мережного принципу в єдиний комплекс. Розглядаючи САПР, в окремі групи виділяють: програмне забезпечення (тексти програм на МНІ), інформаційне (бібліотеки, каталоги на МНІ), лінгвістичне забезпечення (алгоритмічні мови, термінологія).

4) Організація роботи САПР. Ця задача вирішується за допомогою методологічного й організаційного забезпечення.

Методологічне забезпечення –це документи, що відбивають технологію, склад, правила добору й експлуатації ЗОТ.

Організаційне забезпечення - це положення, посадові інструкції, штатні розклади й інші документи, що встановлюють організаційну структуру і функції підрозділів, порядок їхньої взаємодії в умовах функціонування САПР.

3 Вимоги, що пред'являються до САПР

Створювана САПР повинна забезпечувати [17]:

- необхідну продуктивність (допустимо 100 ВІС на рік);

- розмаїтість форм взаємодії проектувальника із САПР, включаючи інтерактивне;

- гнучкість (інваріантність) структури;

- можливість нагромадження досвіду в системі;

- можливість одночасного проектування декількох виробів;

- незалежність запровадження в дію й експлуатації окремих підсистем, які в якості складових утримує САПР;

- інформаційну погодженість підсистем і пакетів прикладних програм для різних етапів проектування.

Перераховані властивості можна реалізувати в САПР, при побудові якої враховувалися наступні принципи:

- комплексність - тобто повинні вирішуватися задачі по всьому циклу проектування (від складання ТЗ до одержання інформації на МНІ);

- сумісність автоматичного, автоматизованого і традиційного режимів проектування (загалом у циклі

проектування утримуються задачі, які підлягають формалізації, частково формалізуються і не формалізуються зовсім);

- однократність опису багаторазово використовуваних даних і проектних процедур у загальному банку даних. До БД доступ здійснюється з різних підсистем і прикладних програм;

- модульність - реалізація окремих підсистем у виді функціонально самостійних модулів з наступною їхньою заміною в міру розвитку й удосконалювання складові системи;

- мультидоступа - система колективного доступу.

Структурна схема типової САПР показана на рисунку 1.

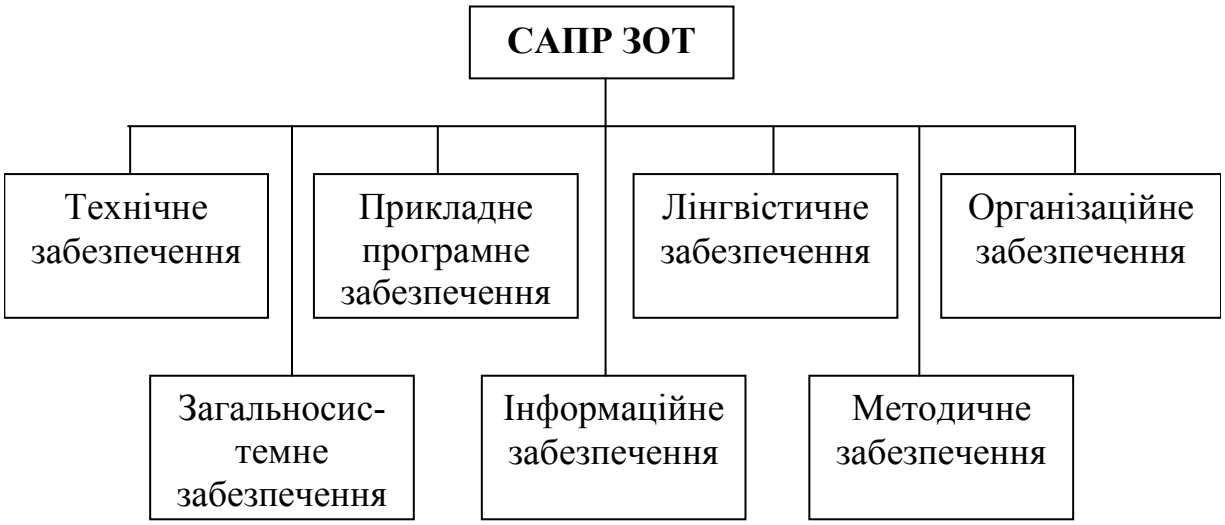


Рисунок 1 - Структурна схема типової САПР

Прикладне програмне забезпечення – це конкретні програми, які розроблені всіх підсистем системи і включають опис наступних процедур:

- проектування алгоритму функціонування і структури побудови схеми виробу, логічних і функціональних схем, принципових електричних схем, топології друкованих плат;

- синтез контрольних тестів;

- підготовки інформації на МНІ для програмно керованого технологічного устаткування.

Технічне забезпечення - це сукупність ЕОМ і периферійних засобів для введення, збереження, переробки, виведення даних і виготовлення машинними методами конструкторської і технічної документації.

Загальносистемне програмне забезпечення - це сукупність системних програмних засобів, що забезпечують виконання прикладних програм, тобто частина програмного забезпечення, призначеного для планування й організації процесу введення, виведення, підготовки і налагодження програм, розподілу ресурсів, призначених операційною системою (ОС). ОС складається з програми 2-х груп:

- обробники систем програмування, що складають систему підготовки програм (зовнішнє програмне забезпечення);

- керуючих, що утворять групу програм для внутрішнього використання системою (внутрішнє програмне забезпечення.)

Інформаційне забезпечення - це накопичення, документування, збереження і видача необхідної для роботи інформації, а саме - БД, організовані для роботи в автоматичному режимі. На цьому етапі можливе створення бази знань проектувальника, БД інструментів, БД технологічних операцій, тощо.

Лінгвістичне забезпечення - це сукупність:

- мов високого рівня: СІ, С++, DELHI і так далі;

- вхідних мов - опис об’єктів проектування;

- мов систем керування БД (СУБД- створення, редагування й експлуатація БД);

- вихідних мов - висновок результатів проектування на МНІ і паперових носіях інформації.

Методологічне забезпечення - документи, що описують маршрут проектування. Містять правила й інструкції з використання програмного забезпечення.

Організаційне забезпечення - перелік документів, що регламентують взаємодію і функції підрозділів, що виступають розроблювачами і користувачами САПР, а також матеріально - технічне забезпечення САПР.

4 Реалізація структур САПР

Застосування ЕОМ повинне охоплювати всі стадії створення виробів нової техніки від зародження задуму до виготовлення та випробування дослідного зразка.

Стадії розробки ВЕТ показані на рисунку 2 [6].

В даний час ведуться роботи із взаємної інтеграції систем АСТПП, САПР, АСУТП і ін., а також по удосконалюванню кожної з них. Найбільші успіхи досягнуті за останні 2-3 десятиліття в області створення САПР ІС, ЗВІС і ВІС. Технічна реалізація загальної структури САПР може мати різні форми в залежності від типу ЕОМ.

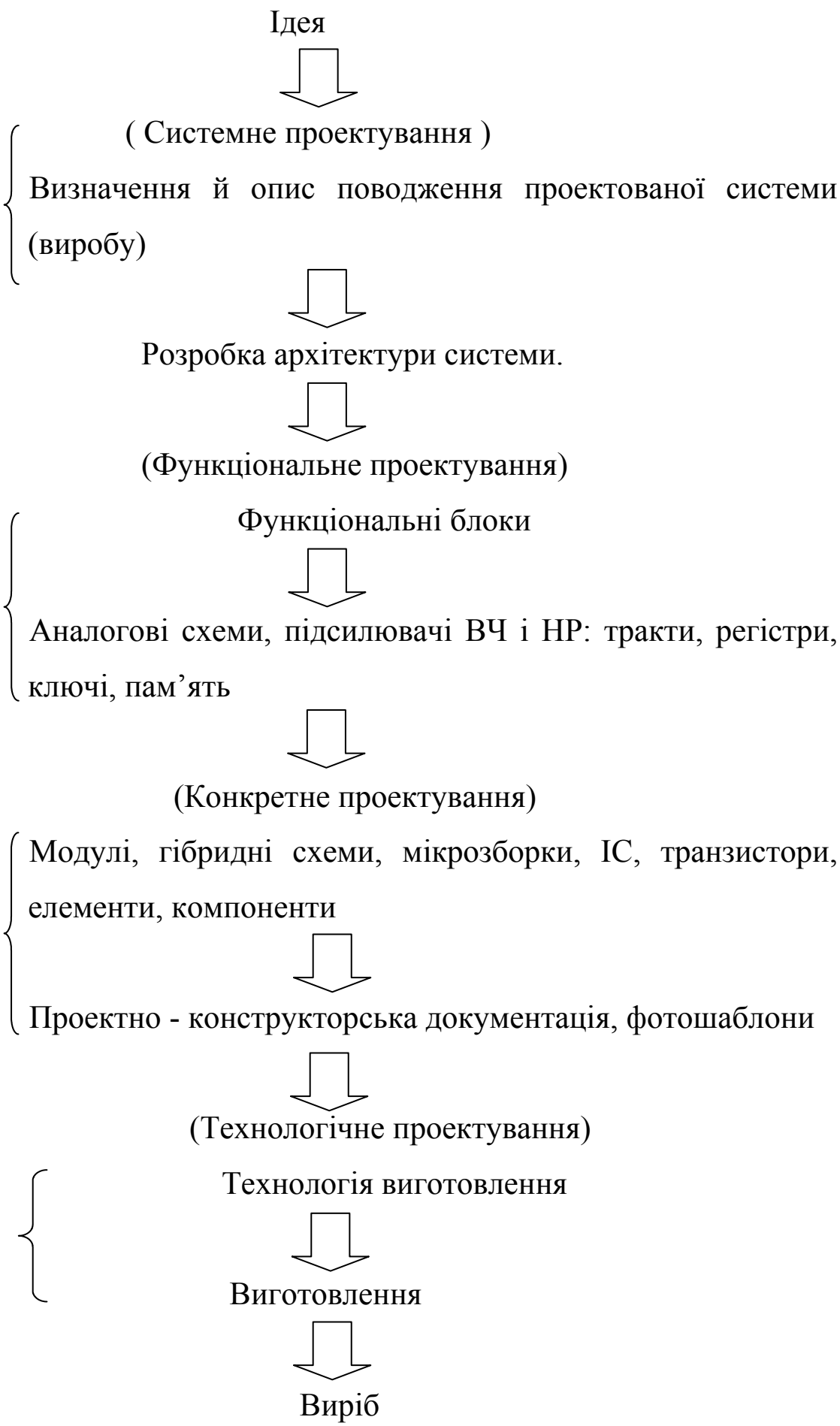


Рисунок 2 - Стадії проектування ВЕТ

Момент появи ІС можна вважати часом переходу від класичних аналітичних методів розрахунку електричних схем до машинних методів розрахунку. Етапи розвитку ІС відповідають етапам розвитку методів проектування.

Ранні стадії розвитку автоматизованого проектування: ЕОМ використовувалися тільки в якості прискорювачів розрахунків аналітичних виразів.

В середині 60-х років визначилась тенденція створення програмного забезпечення з залученням чисельних математичних методів, які і на даний час використовуються при розробці фактично усіх типів САПР. Причому в цих САПР, що продовжують розвиватися по дійсний час, можна виділити загальну характерну рису - вони зберігають структуру окремих програмно - орієнтованих програмних модулів. Це обумовлюється тим, що САПР складається з різних проблемно-орієнтованих програмних модулів (ПРОПМ), створених різними розроблювачами з багатьох організацій.

Для ефективного рішення широкого спектра задач проектування, дані ПРОМП зазвичай об’єднуються в інтегровані САПР (ІНСАПР), які є найбільш поширеним різновидом САПР на даний час. Інтегрування проводиться по числу розв’язуваних проблем, по числу і можливостям підключення ПРОПМ.

Поява ЗВІС і ВІС замовленого і напівзамовленого типу [1], [15] послужило каталізатором для створення іншого різновиду САПР - систем наскрізного автоматизованого проектування (ССАПР). До характерних рис ССАПР віднесемо наступне: єдина високого рівня мова опису об’єктів

проектування, єдина мова опису завдань і конвеєрна обробка інформації. Найбільше застосування вони можуть знайти на підприємствах, які займаються розробкою та виготовленням ЗВІС і ВІС.

Зазвичай, розроблювачі бажають володіти не системою автоматизованого проектування САПР, а системою автоматичного проектування (САВПР), або арсенід кремнієвим компілятором, коли на вхід САВПР подається ТЗ (мовою високого рівня), а на виході системи користувач може одержати комплект конструкторсько-технологічної документації (КТД) на виготовлення ЗВІС і ВІС. Розглянемо короткий опис процесу проектування ОП в умовах діючої САПР (рисунок 2) [6].

Етап системного проектування визначає: функціональні характеристики майбутнього виробу (об’єкту проектування), призначення і структуру вхідних блоків, формується частка ТЗ для проектування наступних етапів.

Далі проект уточнюється з урахуванням електричних характеристик при механічному і тепловому впливі; конструкторсько-топологічне проектування дозволяє цілком визначати конструкцію об’єкта, який підлягає проектуванню, та його топологію.

Під верифікацією об’єкта розуміється перевірка параметрів ОП.

Рішення показаних на маршруті задач, відбувається шляхом створення САПР, побудованих з використанням наступних ідей:

- глобальних досліджень простору проектних рішень;

- інтерактивних засобів проектування на основі дослідження локального простору проектних рішень;

- синтезаторів логічних матриць на основі базової технології;

- кремнієвих компіляторів;

- систем моделювання процесів творчої діяльності людини на основі штучного інтелекту.

З безлічі нових процесів творчої діяльності людини найбільша увага розробниками автоматизованих систем проектування приділяється кремнієвим компіляторам (КРЕМКОМ - це різновид САВПР).

Сучасні САПР ВЕТ та РЕА розвиваються в напрямку усе більшої інтелектуалізації і, у кінцевому рахунку, стануть цілком автоматичними.

МЕТОДОЛОГІЯ ПРОЕКТУВАННЯ КС

1. Визначення та суть інженерного проектування

2. Методологія проектування

2.1 Визначення та суть методології проектування

2.2 Інформація про виріб по етапах його ЖЦ

2.3 Визначення та суть CALS-технологій

2.4 Призначення і область застосування CALS – технологій

3. Класифікація методологій проектування комп’ютерних систем

1. Визначення та суть інженерного проектування

З точки зору системного підходу [11], у відповідності з ГОСТ 22487 - 77, проектування - це процес складання опису, необхідного для створення ще не існуючого об’єкту, шляхом перетворення його вихідного (первинного) опису в кінцевий опис на основі виконання комплексу робіт пошукового, розрахункового і конструкторського характеру.

Таким чином, суть процесу проектування зводиться до наступних основних моментів:

1. Проектування технічного об’єкту пов’язане із створенням, перетворенням та представленням у прийнятій формі образу цього об’єкту як числової моделі.

2. Образ об’єкту проектування або його складових частин може створюватись в уяві людини в результаті творчого процесу.

3. Образ об’єкту проектування або його складових частин може генеруватися за допомогою заздалегідь чітко означених алгоритмів в процесі взаємодії людини та ЕОМ.

2. Методологія проектування

2.1 Визначення та суть методології проектування

Методологія будь-якої діяльності – це вчення про структуру, логічну організацію, методи і засоби цієї діяльності [11].

Методологія проектування полягає в тому, що можливість проектування складних об’єктів обумовлена використанням ряду принципів:

- декомпозиції і ієрархічності опису об’єктів;

- багатоетапності та ітераційності проектування;

- типізації та уніфікації проектних рішень і засобів проектування.

Принцип декомпозиції передбачає структуризацію (розбиття) уявлень відповідного рівня опису об’єкта проектування на складові частини з метою їх роздільного проектування з врахуванням погодження рішень, що приймаються в результаті виконання окремих проектних процедур при завершенні кожного окремого етапу.

Принцип ієрархічності передбачає структуризацію уявлень про об’єкти проектування і їх складові частини за ступенем конкретизації і деталізації опису з метою послідовного нарощування складності опису об’єкту в поєднанні з декомпозицією.

Багатоетапність проектування передбачає розподілення процесу проектування в часі на:

- стадії;

- етапи;

- проектні процедури та операції.

Основними задачами методології проектування є:

- зменшення числа ітерацій (наближень) в процесі проектування;

- зниження затрат часу та вартості розробки проектів;

- підвищенні надійності та якості проектів.

Оскільки САПР припускає використання багатьох методів, то виникає необхідність у вживанні строгих означень складових процесу проектування [21]. Розглянемо цей аспект більш детально.

Технічну систему, яка підлягає проектуванню, можна розглядати з трьох сторін (у філософії це називається модуси станів):

1) як виріб;

2) як пристрій;

3) як процес.

Виріб: складальні одиниці (СО) і деталі (умовно монолітні деталі – МД). Це результат виготовлення і збірки попредметної декомпозиції, який допускає структуризацію (розбивання) уявлень відповідного рівня опису об’єкта на складові частини з метою їх роздільного проектування з врахуванням погодження рішень, що приймаються.

Основними задачами методології проектування є:

- зменшення числа ітерацій (наближень) в процесі проектування;

- зниження затрат часу та вартості розробки проектів;

- підвищенні надійності та якості проектів.

Опис об’єкту проектування по складності повинен бути погоджений з можливостями людини до сприйняття і з можливістю оперувати цими описами в процесі проектування. Однак, виконати ці вимоги в рамках деякого єдиного опису, не розбиваючи його на частини, практично можливо лише для

дуже простих об’єктів. Як правило, необхідно оструктурити опис об’єкту, що проектується, на відповідні розчленування (декомпозиції) уявлень про нього на ієрархічні рівні і аспекти.

2.2 Інформація про виріб по етапах його ЖЦ

Весь об’єм інформації про виріб можна розподілити по етапах його життєвого циклу [13].

1. Конструктивні дані про виріб (КД) – сукупність інформаційних об’єктів, що породжуються в процесі проектування виробу. Містять відомості про склад виробу, про геометричні моделі виробу, про відносини компонентів в структурі виробу, про допуски на виготовлення деталей і т.д.

2. Технологічні дані про виріб – сукупність інформаційних об’єктів, що породжуються на стадії технологічної підготовки виробництва і асоціюються у розробника з конструкторськими даними про виріб. Утримують відомості про способи виготовлення і контролю виробу і його компонентів, опис маршрутних і операційних технологій, норми часу і витрати матеріалу і т.д.

3. Виробничі дані про виріб – сукупність інформаційних об’єктів, що породжуються в процесі виробництва, і асоціюється з КД. Це відомості про статус конкретних екземплярів виробу і його компонентів у виробничому циклі (серія, номер серії, дата виробництва, місце зберігання).

4. Дані про якість виробу – сукупність інформаційних об’єктів, що породжуються при виконанні всіх видів контролю, що асоціюється з КД. Це інформація про ступінь відповідності

конкретних екземплярів виробу і його компонентів заданим технологічним вимогам, вимогам стандартів і інших нормативно-технічних документів.

5. Логістичні дані – сукупність інформаційних об’єктів, що породжуються в процесі проектування і виробництва. Це дані, необхідні для інтегрованої логістичної підтримки виробу на продуктивних стадіях життєвого циклу виробу.

6. Експлуатаційні дані про виріб – сукупність інформаційних об’єктів, що утримують необхідні дані для організації обслуговування, ремонту і інших дій, що забезпечують працездатність виробу.

Необхідність ув’язки величезної кількості різнорідної інформації викликало до життя нову інформаційну технологію - CALS-технологію.

2.3 Визначення та суть CALS-технологій

Розшифровка CALS в залежності від часу її використання [14]:

- 1985 рік – Computer Aided of Logistics Support (Автоматизовані логістичні системи);

- 1988 рік – Computer Aided Acquisition and Support (Автоматизовані постачання і підтримка);

- 1993 рік – Computer Aided Acquisition and Life-Cycle Support (Автоматизація безперервних постачань і життєвого циклу);

- 1995 рік – Commerce At Life Speed (Бізнес у високому темпі).

Отже, під CALS-технологією слід розуміти принципово нові комп'ютерні системи електронного опису процесів розробки, комплектації, виробництва, модернізації, збуту, експлуатації, сервісного обслуговування і утилізації продукції військового, цивільного і подвійного призначення.

2.4 Призначення та область застосування CALS-технологій

CALS-технології призначені для застосування в різних областях [3]:

- у виробництві промислової продукції;
- в банківській діяльності;
- в охороні здоров'я;
- в будівництві і т.д.

Для забезпечення взаєморозуміння розробників, постачальників матеріалів і комплектуючих виробів, виробників і споживачів продукції, що застосовують системи електронного обміну даними, розроблений комплекс міжнародних стандартів по CALS-технологіях.

Технології, що існують сьогодні в промисловості, відносяться до певних етапів ЖЦ виробу (конструювання, розробка технології, планування виробництва і т.д.). Але можливість інформаційної взаємодії між ними відсутня. Виникаючі при цьому непродуктивні витрати західними аналітиками оцінюються для в десятки мільйонів доларів в рік.

Впровадження міжнародних стандартів по CALS-технологіях дозволяє інтегрувати в одну систему комплекс

матеріальних і інформаційних потоків, що існують на всіх етапах життєвого циклу.

Концепція CALS-технологій на першому етапі її розробки полягала в уніфікації і об'єднанні різнотипних комп'ютерних мереж промислових корпорацій з метою створення глобальної системи закупівель і матеріально-технічного постачання, використовуваної при створенні складної машинотехнічної продукції, зокрема: озброєння і військової техніки.

В ході виконання програми реалізації CALS-технологій її первинний задум істотно трансформувався і в даний час перетворився на інструмент комп'ютерного проектування, виробництва, постачань, експлуатації складних технічних виробів, а також - профілактичних і ремонтних робіт в процесі їх експлуатації.

Одна з основних ідей CALS – це можливість включення опису всіх видів виробів в єдину загальну структуру, що допускає обробку різних типів даних і породжених похідних описів властивостей виробів в цій єдиній структурі.

В даний час в CALS виділяють наступні напрями:

- методи аналізу процесів бізнесу;
- методи і засоби паралельного проектування;
- технології логістики;
- практичне використання технологій Інтернет;
- електронна документація на виріб;
- інформаційна безпека;
- уніфікована модель виробу від проектування до утилізації (ISO 10303-STEP);

- юридичні питання інформаційної взаємодії підприємств.

Цілі використання CALS-технологій:

- скорочення витрат на реалізацію ЖЦІ в цілому;

- підвищення ефективності і скорочення витрат в процесах бізнесу;

- підвищення конкурентоспроможності і ринкової привабливості вироблюваної продукції;

- створення передумов для збереження і розширення ринків збуту.

3 Класифікація методологій проектування комп'ютерних систем

Останніми роками спостерігається широкомасштабне застосування обчислювальної техніки в різних сферах діяльності людини. Тому найгостріше встала проблема розробки програмних систем, що автоматизують діяльність розробників КС з метою підвищення її ефективності.

Створення сучасних систем проектування неможливе без використання методологій і технологій їх проектування [16], оскільки їх розробка представляє тривалий, трудомісткий і наукомісткий процес, що вимагає великих матеріальних і фінансових витрат. Доречно згадати тлумачення слова «Технологія»: «Сукупність виробничих методів і процесів у певній галузі виробництва, а також науковий опис способів виробництва». Таким чином, в області створення під терміном «Технологія» розуміється сукупність методів і процесів створення систем проектування КС. Із-за великої складності

процесу їх розробки, методи проектування зазвичай відокремлюються від технологій їх створення, утворюючи методології розробки КС. Тому в подальшому викладі вони розглядатимуться також окремо.

Створено декілька класифікацій методологій і технологій розробки КС. Відповідно до однієї з них вони розділяються залежно від науково-практичних напрямів, у рамках яких виникли. На даний час сформувалися наступні науково-практичні напрями, що займаються питаннями створення методологій і технологій проектування КС [21]:

- інформаційна інженерія (Information Engineering);
- штучний інтелект (Artificial Intelligence);
- зворотне проектування (Re - engineering);
- реінжиніринг бізнес-процесів (Business Process Reengineering);
- багатоагентні системи (Multi - Agent Systems);
- управління знаннями (Knowledge Management);
- промислова інженерія (Industrial Engineering);
- управління якістю (Total Quality Management).

Приведемо їх коротку характеристику. В середині 90-х років в методологіях і технологіях розробки КС сталася зміна однієї з парадигм: програмна інженерія (Software Engineering) змінилася на інформаційну інженерію (Information Engineering).

Інформаційна інженерія є сукупністю методологій і програмних інструментальних засобів, які підтримують створення програмних та комп’ютерних систем, що автоматизують діяльність людини.

Головною відміною особливістю інформаційної інженерії від програмної інженерії є наявність методів і програмних інструментальних засобів, що підтримують етап стратегічного планування життєвого циклу програмного забезпечення. На етапі стратегічного планування здійснюється обстеження діяльності організації з метою підвищення ефективності праці співробітників.

Для цього аналіз організації проводиться на трьох рівнях: макрорівні; мікрорівні; рівні організації.

На макрорівні здійснюється аналіз політичної обстановки, економічного стану і технічної політики, які впливають на вибір сфер діяльності організації. На мікрорівні здійснюється аналіз ринкових стосунків між організацією, споживачами і конкурентами.

На рівні організації здійснюється аналіз внутрішнього стану організації: організаційна структура, виробництво продукції, фінансове положення, професіоналізм кадрів і тому подібне.

Стратегічний аналіз діяльності організації представляє трудомісткий тривалий процес, ціна помилок на якому дуже висока, оскільки вони можуть привести до краху організації.

CASE -технології (Computer - Aided Software Engineering) [4], [5]виникли у рамках програмної інженерії і дозволяють значно скоротити час проектування КС, підвищити їх якість, а також значно зменшити витрати на їх створення. Науково-практичний напрям «CASE-технології» займається питаннями створення методологій проектування програмних систем і

програмних інструментальних засобів їх підтримки (CASE - засобів).

Найширше вживаними методологіями є представники двох класів методологій: структурних і об'єктно-орієнтованих.

Структурні методи проектування систем з'явилися у кінці 70-х років. Їх створення пов'язане з іменами Йордона, Джексена, Якобсона і багатьма іншими. Слід зазначити, що Йордона є розробником першого CASE – засобу, створеного в 1979 р., в якому на основі побудованих діаграм генерується програмний код на мові Пекла. До теперішнього часу створений чисельний ряд CASE - засобів, що підтримують структурні методи. До нього відносяться CASE - засоби: BPWin, Idef, Silverrun і багато інших.

Останніми роками найбільш перспективною методологією для створення КС визнана об'єктно-орієнтована методологія, одним з основоположників якої є Г.Буч. CASE - засоби, що підтримують цю методологію, широко застосовуються у фірмах в США, Японії і в інших провідних країнах світу. В нашій країні більш широкого використання набули об'єктно-орієнтовані методи, являються Rational Rose Enterprise Edition 2000/2002 (фірми Rational Software Corporation), Oracle Developer Suite 2000 (фірми Oracle) та ін. Слід зауважити, що Г.Буч і Дж.Рамбоух є творцями мови UML (Unified Modeling Language), що з'явилася в 1994 р., і яка покладена в основу сучасної мови програмування Rational Rose Enterprise Edition.

Перспективним підходом до створення CASE - засобів являється застосування в них інтелектуальних методів, що

обумовлено наступними чинниками. На етапах створення програмних систем розробникам доводиться працювати з неповною, нечіткою, неточною, суперечливою інформацією. Створені CASE – засоби на основі використання інтелектуальних методів дозволяють здолати труднощі, що виникають у розробників, швидшими темпами і з меншими трудовитратами. Бурхливий ріст робіт, що стосуються питань використання інтелектуальних методів в CASE - засобах, став основою для появи науково-практичного напряму у рамках інформаційної інженерії, що дістав назву «Knowledge Based Software Engineering (KBSE)».

Одним з основних результатів науково-практичного напряму, що займає питаннями створення інтелектуальних систем (ІС), являються інтелектуальні методології і технології їх проектування. Якщо в період становлення цього напряму вважалося, що такі методології і технології базуються на моделях представлення знань і їх механізмів виведення, то останніми роками спостерігається тенденція до їх інтеграції. Відзначимо, що найширше використовуваними методологіями є методології проектування ІС, засновані на моделях представлення знань, м’яких обчислень, що об’єднали нечітку логіку, нейротехнології і генетичні алгоритми, а також, слід згадати про синергетичний підхід, що набирає силу і стає все більш і більш популярним.

Методи зворотного проектування і програмні інструментальні засоби його підтримки спрямовані на оптимізацію характеристик створених програмних систем і

забезпечення їх «стикування» з існуючими в організації програмними системами.

Одночасно з інформаційною інженерією з’являється напрям менеджменту, що дістав назву - реінжиніринг бізнес-процесів. Ключові моменти цього напряму будуть розглянуті при приведенні 2-ої класифікації методологій і технологій проектування програмних систем, під впливом якого він і сформувалася.

Багатоагентні методології і технології розробки КС, що завоювали популярність у кінці 90-х років, визнані, як одні з найбільш перспективних.

Ще одні з найбільш перспективних методологій і технологій створення КС являються методології і технології їх побудови як систем управління знаннями (СУЗ), де під управлінням знаннями розуміється методологія, що включає комплекс формальних методів, що охоплюють :

- пошук і витягання знань з живих і неживих об’єктів (носіїв знань);
- структуризацію і систематизацію знань (для забезпечення їх зручного зберігання і пошуку);
- аналіз знань (виявлення залежностей і аналогій);
- оновлення (актуалізацію) знань;
- поширення знань;
- генерування нових знань.

Виробнича інженерія [11], що виникла в середині XX століття, займається управлінням і організацією виробництва. У ній найширше вживаними методологіями і технологіями є

JIT (Just - in - time - точно вчасно), OPT (Optimised Production Technology - оптимізаційна технологія виробництва), CIM (Computer Integrated Manufacturing - інтегровані виробництва на основі обчислювальної техніки), CALS (Continuous Acquisition and Life Circle Support - підтримка безперервного життєвого циклу продукції), ERP, MRP (Material Requirements Planning - планування потреб в матеріалах), MRP II (Manufacturing Recourse Planning - планування ресурсів виробництва), CAD/CAM/CAE і так далі. Розроблений і впроваджений ряд ERP-систем.

Технологія розробки систем якості (Total Quality Management) базуються на концепції управління якістю, документованою в стандартах ISO 9000. Слід зазначити, що стандарт ISO 9000 є серією стандартів 9000, 9001, 9002, 9003, 9004, в якій ISO 9001 є якнайповнішим стандартом, що специфікує модель забезпечення якості на усіх етапах життєвого циклу товару/послуги. [4].

Останніми роками стираються межі між виділеними класами методологій, оскільки здійснюється проникнення методів проектування систем одних методологій в інші. Так, наприклад, інтелектуальні методи застосовуються в інформаційній інженерії (розробка CASE - систем, заснованих на знаннях) , в реінжинірингу бізнес-процесів (використання моделей міркувань, заснованих на прецедентах (Case - based reasoning), в багатоагентні технології (проектування інтелектуальних агентів). Без їх використання неможливе створення систем управління знаннями.

Друга класифікація методологій і технологій створення програмних систем, як уже згадувалося, склалася під впливом реінжинірингу бізнес-процесів (БПР).

Нині досить велике число організацій, включаючи університети, мають організаційні форми і процедури функціонування, що не забезпечують конкурентоздатності організації і не сприяючі її розвитку. На основі методів БПР вони мають бути кардинальним чином перебудовані. Хаммер і Чамплі, винахідники терміну БПР, визначили БПР як «фундаментальне переосмислення і радикальне перепроектування ділових процесів для досягнення різких/стрибкоподібних поліпшень у вирішальних, сучасних показниках діяльності компанії, таких як вартість, якість, сервіс і темпи». БПР - є сукупність методів і програмних інструментальних засобів, призначених для кардинального поліпшення основних показників діяльності, шляхом моделювання, аналізу і перепроектування існуючих бізнес-процесів.

Одним з важливих класів ІС, вживаних в різних сферах діяльності людини, являються інтелектуальні системи підтримка ухвалення рішень (СППР), що допомагають людині при управлінні ними складними системами і процесами. Перед розробниками СППР на початкових етапах їх створення постає завдання моделювання поведінки складних об’єктів управління (ОУ).

При рішенні цієї задачі розробляється модель функціонування ОУ і здійснюється витягання знань з експертів

і осіб, що приймають рішення, і формалізація нормативних документів по управлінню, які представляються в моделях ухвалення рішень. Побудова таких моделей є трудомістким і наукомістким процесом, який не завжди закінчується успішно, оскільки їх створення відбувається зазвичай в умовах неможливості розробки математичної моделі ОУ із-за неповної, суперечливої, нечіткої інформації, що виділяється при описі реальних ОУ.

Збір статистичних даних ОУ, є тривалим процесом а іноді і неможливим із-за складності проведення вимірів характеристик ОУ. Тому одним з перспективних підходів до рішення задачі моделювання поведінки складних об'єктів є використання інтелектуальних методів.

Останніми роками спостерігається різкий ріст кількості робіт, що стосуються логічного підходу до рішення цієї задачі. Великий вклад в цей науковий напрям внесли і вносять зарубіжні логіки, серед яких можна виділити Дж. Маккарті, перші результати якого була отримані в 1963 г. Великий вклад в цей науковий напрям вносить Р. Рейтер, який розробив логічний фундамент для моделювання динамічних систем, створив мову логічного програмування GOLOG (Algol in Logic), реалізував різні версії цієї мови: послідовний GOLOG, паралельний GOLOG, тимчасовою GOLOG, реактивний GOLOG. Семантика мови GOLOG описується численням предикатів другого порядку. Синтаксис мови нагадує синтаксис мови Prolog. Також тут можна згадати Р.Міллера і М.

Шехенена, що розробили теорію зчислення подій, і багатьох інших зарубіжних логіків.

Українські вчені також вносять великий вклад в створення логічних методів моделювання складних об'єктів, вершиною робіт в якому став підхід семіотичного моделювання, запропонований Д.А.Поспєловим, надаючий математичний базис для побудови ІС якісно нового рівня. Так, на зміну формальної системи і її часткових модифікацій приходить семіотична система, що є фундаментом для створення семіотичних моделей, що дозволяють адекватно описувати сучасні складні проблемні середовища.

ОБ'ЄКТ ПРОЕКТУВАННЯ. ПРОЦЕС ПРОЕКТУВАННЯ

1. Сутність поняття «об'єкт проектування» і поняття «формалізація »

2. Класифікація об'єктів проектування

3. Організація технічного процесу проектування ОП

4. Декомпозиція задач і системний підхід

5. Принципи проектування

1 Сутність поняття "об'єкт проектування" і поняття «формалізація »

Проектування - це процес при якому вихідна інформація про об'єкт проектування на рівні ідеї перетвориться в комплект КТД для його подальшого виготовлення за допомогою відповідних технологій [21].

Об’єкт проектування - це не існуюча ще в природі технічна система, що має технічний опис (технічне завдання), яке містить:

- вихідну інформацію (алгоритми функціонування, алгоритми процесу проектування);

- кількісні вимоги до функціональних параметрів об’єкта проектування.

Проектування відповідає інформаційному процесу, у якому здійснюється перетворення вхідної інформації про проектований об’єкт з урахуванням стану знань у розглянутій області, у вихідну інформацію - комплект КТД , складеної з усіма вимогами діючих ДСТУ, і утримує проектне рішення або результати проектування. З поняттям «об’єкт проектування» в САПР нерозривно зв’язане поняття «формалізація об’єкта проектування». Формалізація - від латинського formalis - складений за формою. У математиці - це метод подання теорії у вигляді обчислень. По суті, процес формалізації полягає в заміні всіх змістовних термінів символами, а всіх змістовних тверджень - групою символів (формулою). Здійснюється шляхом виявлення і перебудови структури теорії, унаслідок чого вона здобуває вигляд ланцюга формул, де кожна наступна логічно випливає з першої або попередньої.

Цей принцип ліг в основу формалізації як об’єктів проектування, так і процесу проектування.

2. Класифікація об'єктів проектування

Насамперед була проведена класифікація об'єктів проектування по ознаках [21],[13] По фізичних принципах роботи:

- радіоелектронні;
- механічні;
- обчислювальні процеси;
- гідравлічні і так далі

За умовами експлуатації:

- космічні;
- наземні;
- морські.

По характеру основних фізичних процесів:

- безперервні;
- дискретні.

Об'єкти проектування можна розділити на вироби і процеси, а процеси - на технологічні й обчислювальні. Оскільки на будь-якому ієрархічному рівні користуються поняттям «система» й «елемент», то можна виділити їхні параметри:

Параметри проектування системи:

- продуктивність;
- вартість;
- габаритні розміри і т.д.
- вихідні характеристики;
- параметри елементів.

Параметри елементів:

а) вихідні - кількісні показники, що характеризують функцію, виконувану об’єктом. Наприклад: транзистор - коефіцієнт передачі по струму, САПР- кількість розроблених друкованих плат ;

б) внутрішні - це параметри складових, з яких складається об’єкт, наприклад, для друкованих плат, внутрішнім параметром можуть служити геометричні розміри ЕРЕ, для ВІС - це вихідні параметри транзисторів, конденсаторів, діодів, тощо;

в)зовнішні параметри - це параметри зовнішнього, стосовно проектованого об’єкта середовища. Це - рівень вологості, випромінювання, діапазон температур і так далі.

Загальний взаємозв’язок між вихідними, внутрішніми, і зовнішніми параметрами можна показати в такий спосіб. Уведемо позначення:

- $Y(y_1, y_2,.. y_n)$ - вектор вихідних параметрів;
- $X(x_1,x_2,...x_n)$- вектор внутрішніх параметрів;
- $Q(q_1,q_2,.. q_n)$- вектор зовнішніх параметрів системи.

Тоді:

$$Y=F(X,Q)$$

Конкретний вигляд формули визначається структурою складових об’єкта проектування.

Отримана залежність є справедливою для будь-якого етапу проектування системи автоматизованого проектування, ВЕТ, РЕА, або ЗОТ і є математичною моделлю, представляючи собою одну з формалізованих основних проектних процедур.

Класифікація (формалізація) основних задач на початкових стадіях проектування дозволяє створити САПР пошуку і вибору технічних рішень, до складу якої входять наступні блоки:

- Блок 1: підсистема пошуку аналогів. Під аналогом ОП розуміють реально існуючий об'єкт техніки або не відоме ще технічне рішення, ознаки яких задовольняють технічні вимоги ТЗ. Підсистема пошуку аналогів дозволяє вибрати наступні варіанти:

- Варіант 1: вироби, показники якості яких задовольняють наступні нерівності

$$a_н >= a_{тз} \text{ - знайдено аналог}$$

$$a_н <= a_{тз} \text{ - аналог відсутній;}$$

- Варіант 2: вироби, що містять конструкторські вимоги, технологічні або функціональні ознаки;

- Варіант 3: вироби, що задовольняють вимоги варіанта 1 і варіанта 2.

Підсистема пошуку аналогів дозволяє знайти аналог, необхідний для нової розробки, або визначити його відсутність в базі даних (БД).

- Блок 2: підсистема синтезу технічних рішень (ТР)-1;

- Блок 3: підсистема синтезу фізичних схем;

- Блок 4: підсистема синтезу ТР-2.

Основою САПР пошуку і вибору технічних рішень є наступні модулі (підсистеми) системи автоматизованого проектування:

- пошуково - інформаційна система.

- база даних, орієнтована на реальні вузли, програми, на які в наявності є технічна документація.

Процес проектування реалізується в строго визначеному плані і відповідно до побудованого логічного графіка виконання проекту.

3. Організація технологічного процесу проектування

Процес проектування містить проектні процедури і проектні рішення.

Проектна процедура відповідає формалізованої сукупності дій, що закінчуються ухваленням технічного рішення [21]. Тобто, проектна процедура складається з n - ї кількості елементарних проектних рішень.

Приклад операції, деякі обчислювальні роботи (рішення рівнянь), види підготовки даних, обробка даних, трафік даних в мережі, тощо.

Процедури проектування спираються на мову проектування, що служить засобом лінгвістичного або графічного опису та неоднократного перетворення опису з одного типу моделі в іншу при проектуванні. Наприклад – пристрій уведення графічної інформації (дигітайзер, сканер, фотоплотер, тощо) перетворить графічне зображення ОП у мову опису, зрозумілу для ПК.

Істотну роль грає методологія проектування, що включає основні етапи процесу проектування. Вона повинна бути:

- загально - придатною для широкого кола інженерних задач;

- здатною давати рішення для проектування більш складних систем;

- доступною для вивчення і використання;

- забезпечувати високу якість проектування, надійність.

Методологія проектування базується на загальній теорії систем (наприклад: теорії багаторівневих ієрархічних систем), дискретній математиці, теорії рішень і інших наук. Рішення творчих задач при проектуванні ОП розділяють на евристичні і систематичні.

Евристичні - це такі рішення, коли важлива частина творчого проекту й одержання результату утримуються в мозку людини і не можуть бути логічно отримані з попереднього досвіду проектувальника системи чи конструктора.

Систематичними називаються рішення, одержувані в результаті узагальнення попереднього досвіду і стимуляції мозку людини.

4. Декомпозиція задач і системний підхід

Для логічного процесу творчості у будь-якій сфері діяльності людини характерна декомпозиція - розбивка задачі на складові частини (більш складної задачі на декілька простих) [21].

При системному підході [11], [22], а він не віддільний від САПР, ОП розглядається, як деяка система, що складається з підсистем. Кожна з цих підсистем може бути представлена

підсистемами більш низького рівня. Підсистемами найнижчого рівня є елементи, внутрішня структура яких не представляє інтересу для рішення задач визначеного рівня, однак їхні властивості можуть впливати на систему в цілому.

Кожна система, у свою чергу, є підсистемою системи більш високого рівня, а та - підсистемою системи ще більш високого рівня. Цілком побудована ієрархія системи містить нескінченну кількість систем і підсистем. Однак, при рішенні технічної задачі немає необхідності будувати всю ієрархію, досить включити тільки системи і підсистеми на два рівні вище і нижче, істотно пов’язаних з проектною процедурою.

Сучасне проектування, в процесі якого використовуються ПЕОМ, спирається на відпрацьовану технологію проектування. Це дозволяє упорядкувати інформацію, яка використовується по вертикалі (у відповідності з логічною схемою побудови, проекту) та по горизонталі (у відповідності з системним зв’язком між елементами задачі, що підлягає рішенню).

Як правило, декомпозицію проектних задач виконують по рівням та етапам та рівням проектування.

Декомпозиція технічного об’єкту по рівням рекомендується при проектуванні комплексних систем, які налічують декілька складних систем. Декомпозиція проектної задачі по етапам проектування рекомендується для складних систем дозволяє досягнути рівня деталізації (простих систем).

Можна виділити такі рівні декомпозиції:

- системний – узагальнений опис призначення ОП та його зв’язків з урахуванням тих змін, які ОП внесе в навколишнє середовище;
- архітектурний – опис структури ОП;
- функціональний – опис законів функціонування підсистем ОП;
- конструктивний – детальний вибір та опис всіх елементів ОП,

та етапи декомпозиції:

- формування технічної задачі та визначення напрямку пошуку. На цьому етапі уточнюється ТЗ, виконується постановка задачі, визначається коло пошуку технічного рішення;
- попереднє проектування (пошукове). Вибір та обґрунтування варіанту рішення ;
- ескізне проектування: інженерний синтез (моделювання і оптимізація проектного рішення);
- технічне проектування: виготовлення робочої документації проекту;
- оцінка проекту. Прийняття рішення;
- уточнення рішення задачі: корекція технічної документації по результатах випробувань експериментального зразка;
- прийняття технічного рішення, як це показано на рисунку 2.

До етапів декомпозиції віднесемо наступні проектні процедури:

- формування технічної задачі та визначення напрямку пошуку. На цьому етапі уточнюється ТЗ, виконується постановка задачі, визначається коло пошуку технічного рішення;

- попереднє проектування (пошукове). Вибір та обґрунтування варіанту рішення ;

- ескізне проектування: інженерний синтез (моделювання і оптимізація проектного рішення);

- технічне проектування: виготовлення робочої документації проекту;

- оцінка проекту. Прийняття рішення;

- уточнення рішення задачі: корекція технічної документації по результатах випробувань експериментального зразка;

- прийняття технічного рішення, як це показано на рисунку 1.

Однак, і при декомпозиції по етапах проектування, і при декомпозиції по рівнях проектування однотипність та інваріантність використаних проектних процедур зберігається. Тільки із-за відмінної ступені деталізації проектних рішень, якщо порівнюємо декомпозицію по етапах проектування з декомпозицією по рівнях проектування, використовуються різні методи моделювання, оцінки та вибору проектних рішень.

На архітектоніку процесу проектування впливають також наступні фактори: історично складений досвід проектної організації, творчий почерк генерального конструктора,

особливості конструкції і експлуатаційних характеристик майбутнього вибору і т.д.

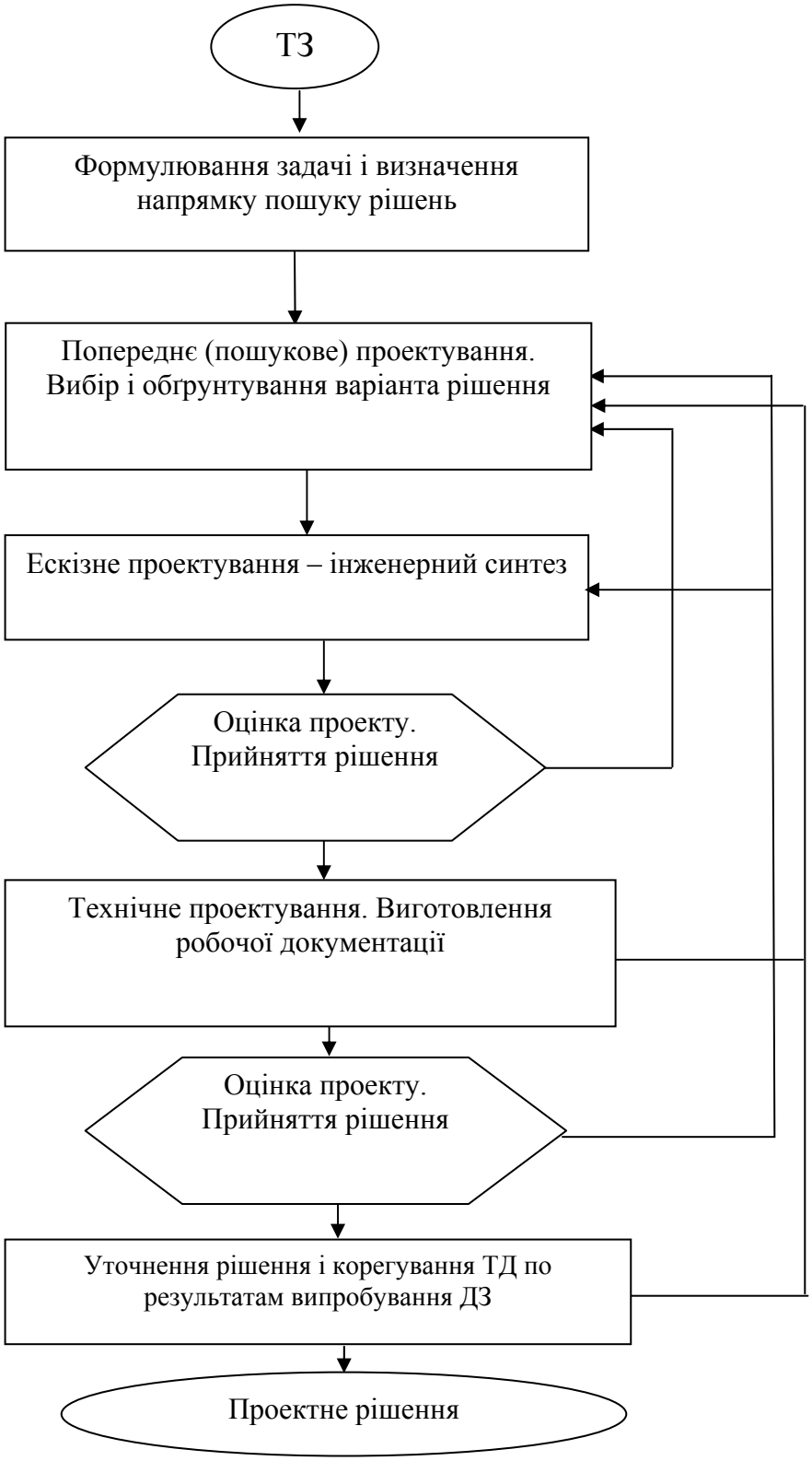


Рисунок 1 - Типовий алгоритм процесу проектування ОП по етапах проектування

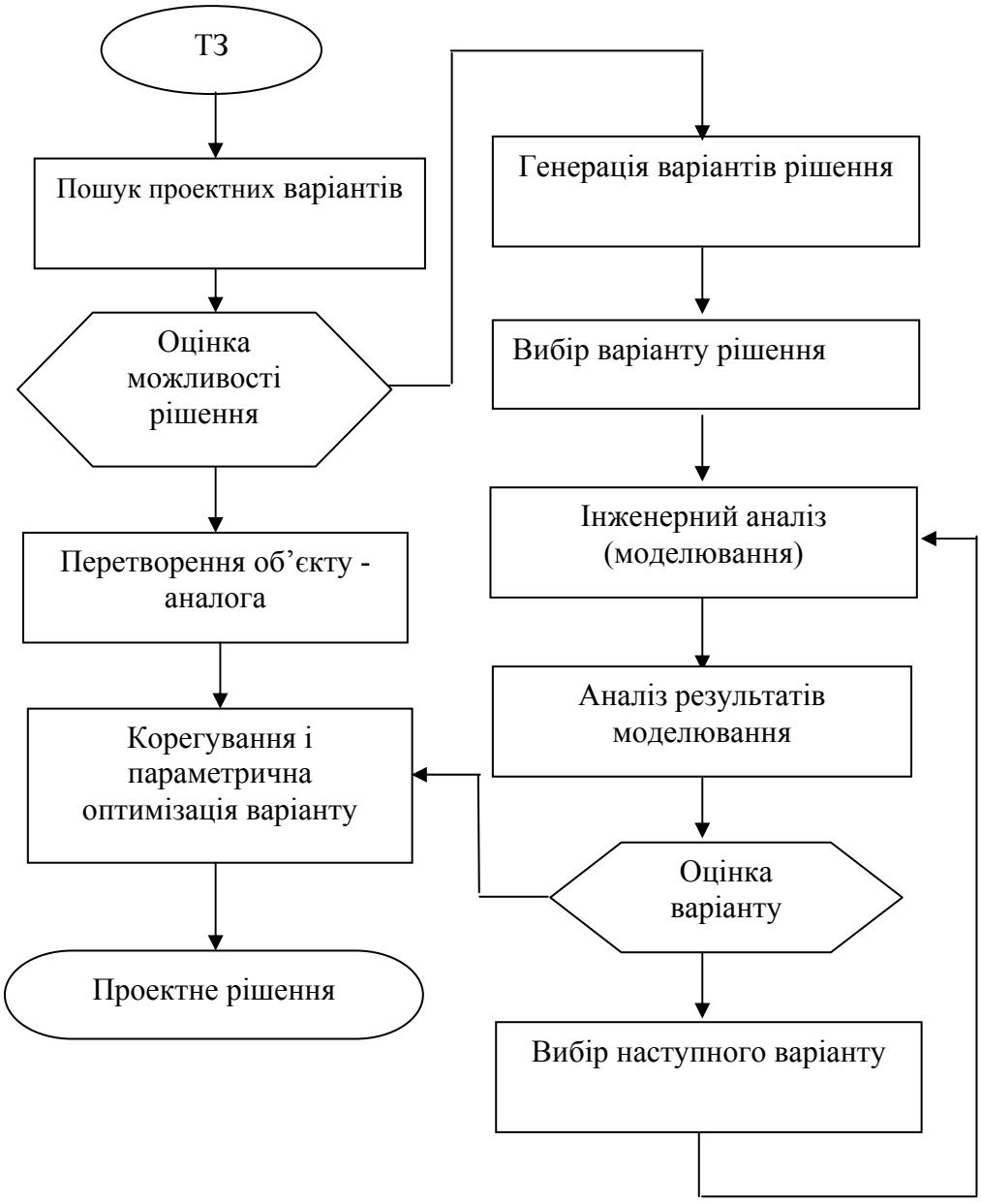


Рисунок 2 - Типовий алгоритм процесу проектування ОП по рівнях проектування

Процес декомпозиції рекомендується виконувати по кількісним параметрам (по числу компонентів на печатній платі) або по ступеню докладного розгляду параметрів майбутнього виробу (макромоделювання регістрів, вузлів і т.д.).

5. Принципи проектування

Їх п’ять:

1) Декомпозиційний - не будемо повторюватися.

2) Ієрархічний - варто додати наступне: це розбиття по ступені подробиці властивостей ОП на рівні. Чим складніший ОП , тим більше рівнів. Наприклад : при проектуванні транзисторів ми маємо тільки один рівень: фізико-технологічний, по якому розраховуються його вихідні параметри (по його внутрішніх і зовнішніх параметрах, по математичній моделі):

$$Y=F(X,Q)$$

Наприклад, проектування однокристальної ЕОМ складається з 5 етапів:

- архітектурно-структурного;
- функціонально-логічного;
- схемотехнічного;
- топологічного;
- фізико-технічного.

3) Ітераційний - перевірка правильності технічного рішення на стадії проектування , тобто до створення дослідного зразка. Тому на кожнім етапі проектування необхідно передбачити послідовне наближення до виконання заданих вимог (за результатами моделювання й оптимізації).

4) Уніфікація задач і основних складових частин ОП.

Основна мета - мінімізація складових ВЕТ і РЕА, необхідних для цілком нової розробки. Так, при розробці РЕА

необхідно застосовувати електрорадіоелементи тільки визначених типів (на які маються відповідні ДСТУ та інші нормативні документи) і т.д.

5) Принцип контролюємості етапів проектування.

Контроль може або бути сполучений з етапом проектування, або виділений з нього (автоматичні системи) - це так називана верифікація. Саме введення при розробці автоматизованих систем покрокової верифікації забезпечує високу якість проектування, а, відповідно, і ефективність САПР будь-якого типу.

Питання організації контролю в процесі проходження ОП по проектних процедурах ми розглянемо більш докладно в наступних лекціях.

ЕТАПИ І РІВНІ ПРОЕКТУВАННЯ

1 Етапи і рівні проектування

Загальна технологія проектування ОП передбачає ряд стадій розробки. Сучасне проектування, у процесі якого використовуються електронні обчислювальні машини, спирається на відпрацьовану технологію проектування. Остання дозволяє упорядкувати в процесі проектування використовувану інформацію по вертикалі (відповідно до логічної схеми побудови проекту) і по горизонталі (відповідно до системного зв'язку між елементами розв'язуваної задачі).

Для логічного процесу творчості характерна декомпозиція – розбиття однієї складної задачі на складові частини [21], [6]. Можна виділити наступні етапи проектування: формулювання задачі, попереднє проектування, ескізне проектування, технічне проектування, прийняття проекту. До рівнів проектування

віднесемо: пошук проектних варіантів, перетворення об’єкта – аналога, корегування і параметрична оптимізація обраного базового варіанта.

Якщо декомпозицію проводити не по етапах, а по рівнях проектування, то однотипність і інваріантність використовуваних процедур проектування зберігається. Ці два види декомпозиції розрізняються лише різною ступеню деталізації і використанням на окремих етапах (рівнях) проектування різних методів моделювання, оцінки і добору проектних рішень. Але рівні декомпозиції рекомендують для проектування більш складних ОП і в тих випадках, коли є імовірність знаходження і перетворення аналога ОП.

Етапи проектування. Типова логічна схема процесу проектування з декомпозицією по етапах надана на рисунку 1.

Початкова стадія проектування - постановка задачі на проектування ОП визначеною вихідним її формулюванням згідно технічного завдання. Як правило, ТЗ на 20-40% неточне, тому необхідно виконати етап уточнення ТЗ для розв’язуваної технічної задачі, а саме - виділення функцій майбутнього об’єкта проектування і виявлення діючих обмежень при його проектуванні.

Оскільки пошук рішень у випадково обраному напрямку не приводить до потрібного рішення, необхідно до процесу проектування ввести спеціальний етап визначення напрямку пошуку. Далі в обраному перспективному напрямку організовується пошук рішень і з безлічі отриманих результатів за допомогою аналізу вибирається кращий.

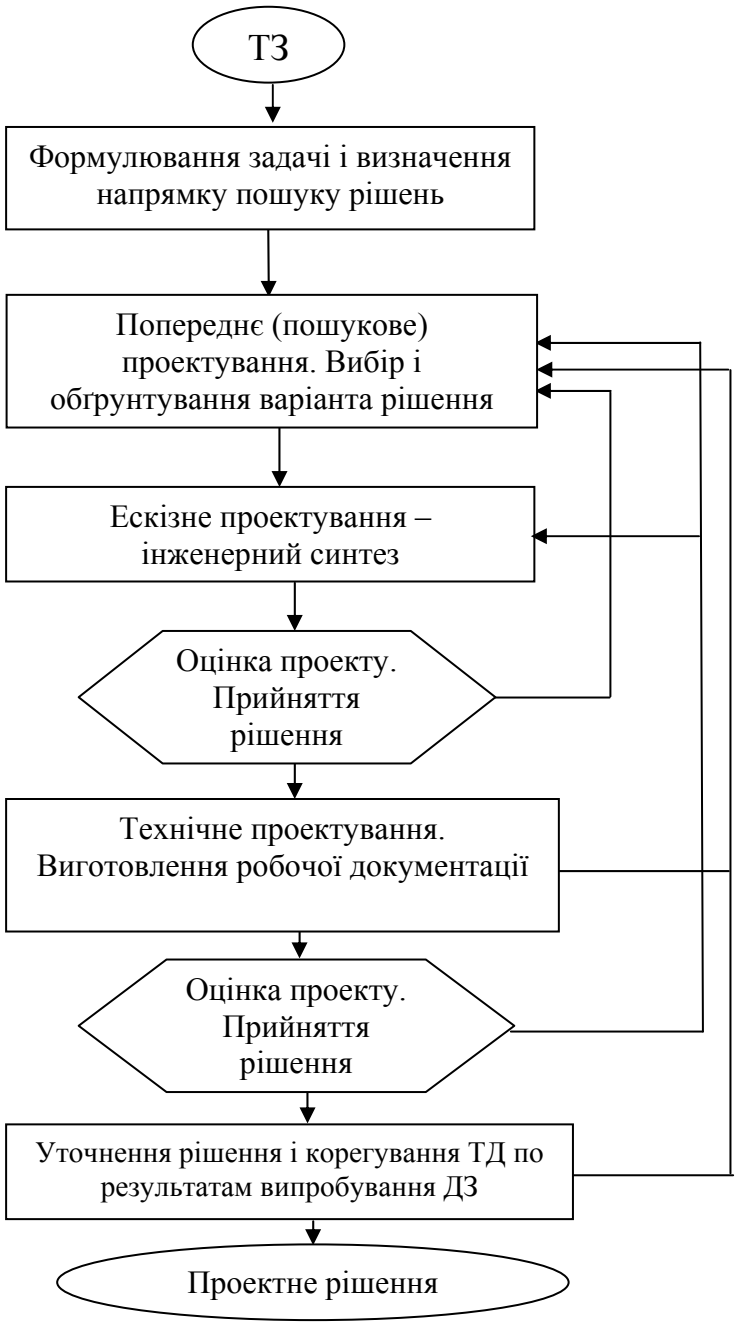


Рисунок 1 – Типова логічна схема процесу проектування

Попереднє проектування починається з вибору структури ОП і матеріально-енергетичних засобів для його реалізації, визначення характеристик ОП і складових його ланок.

Ескізне проектування : визначається напрямок подальшого уточнення і корегування структурної схеми об’єкта, а також проводиться детальний аналіз

використовуваних технічних засобів і виконується їх оптимізація. Для цієї мети використовуються задачі інженерного синтезу й аналізу:

1) синтез - формування принципів реалізації і конкретизації ТР;

2) аналіз - проведення досліджень і їх оцінка.

Оцінки супроводжуються ухваленням рішення.

Однією з основних цілей проектування є оптимізація рішень, тобто - досягнення заданих показників ОП при min витратах. Сутність оптимізації зводиться до відшукування, при накладених обмеженнях, таких структур і значень параметрів ОП, що min (max) задовольняють комплексну ефективність ОП.

Процес оптимізації характеризується дискретним вибором структури і безперервним пошуком напрямку переміни змінних параметрів для поступового наближення до оптимального рішення.

На етапі технічного проектування випускається конструкторська документація (КД) і технічна документація (ТД), необхідна для виготовлення дослідної партії ОП у заводських умовах.

Що загальне для всіх етапів проектування? Поглиблення аналізу й уточнення моделей і, як наслідок, наближення ОП до заданих в ТЗ характеристик.

При цьому моделі від етапа до етапа спадково змінюють одна одну: тестова (ТЗ, математична, інформаційно-графічна (схеми, креслення), макет ОП, експериментальні і дослідні зразки, тощо.

1.2 Рівні проектування

Якщо декомпозицію процесу проектування провести не по етапах попереднього, ескізного і технічного проектування, а по рівнях:

1) системному;

2) архітектурному;

3) функціональному;

4) конструктивному,

то однотипність процедур проектування зберігається, структурна схема ітераційного алгоритму процесу проектування при цьому містить укрупнені проектні процедури, як це показано на рисунку 2 [21]. Від ступені деталізації залежить метод моделювання, оцінки і добору проектних рішень.

Логічні схеми процесів проектування, навіть для того самого класу об’єктів, можуть багато в чому відрізняються. На архітектоніку процесу проектування істотно впливають наступні фактори:

1) історично сформований досвід проектної організації;

2) творчий почерк генерального конструктора;

3) технічна оснащеність проектної організації;

4) особливості конструкції ОП, тощо.

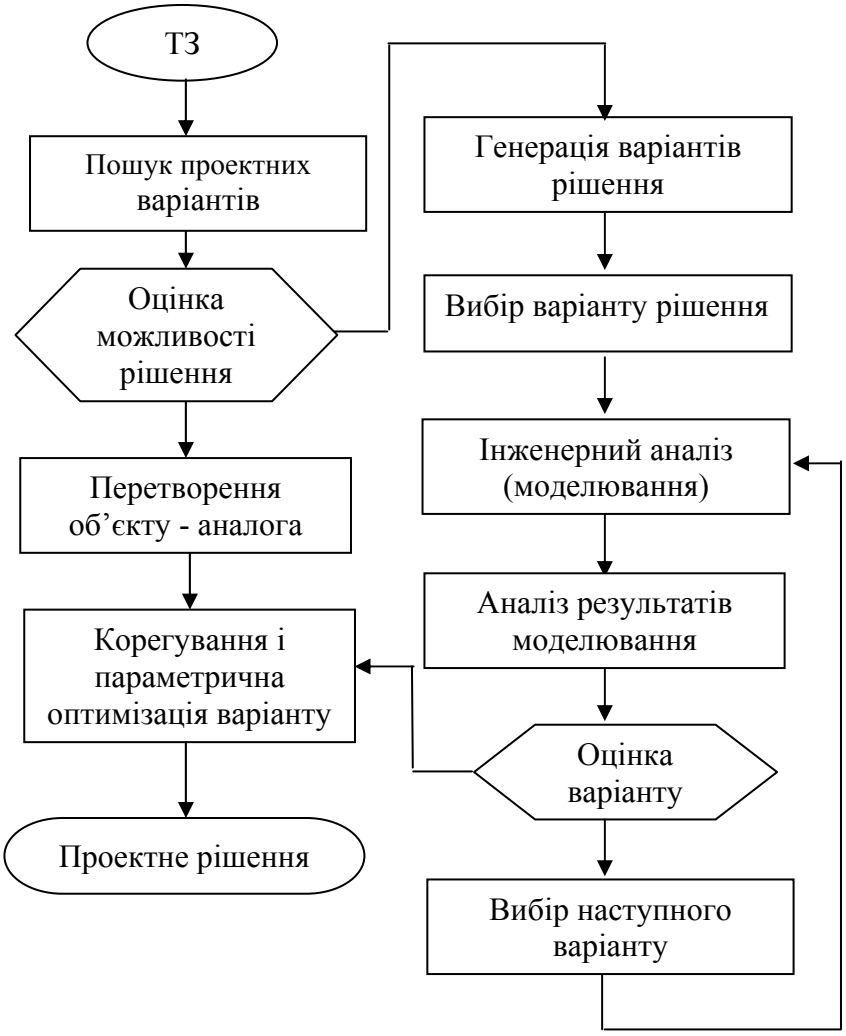


Рисунок 2 - Структурна схема ітераційного алгоритму процесу проектування з декомпозицією по рівнях проектування

1.3 Базові підсистеми САПР

Підсистеми - основні структурні ланки САПР. Підсистема - це виділена по деяких ознаках частина САПР, яка забезпечує виконання деякої закінченої проектної процедури з одержанням відповідної документації і проектних рішень [21].

Розглянуті нами базові етапи і рівні проектування реалізуються підсистемами перетворення інформації, що входять до складу САПР, як це показано на рисунку 3.

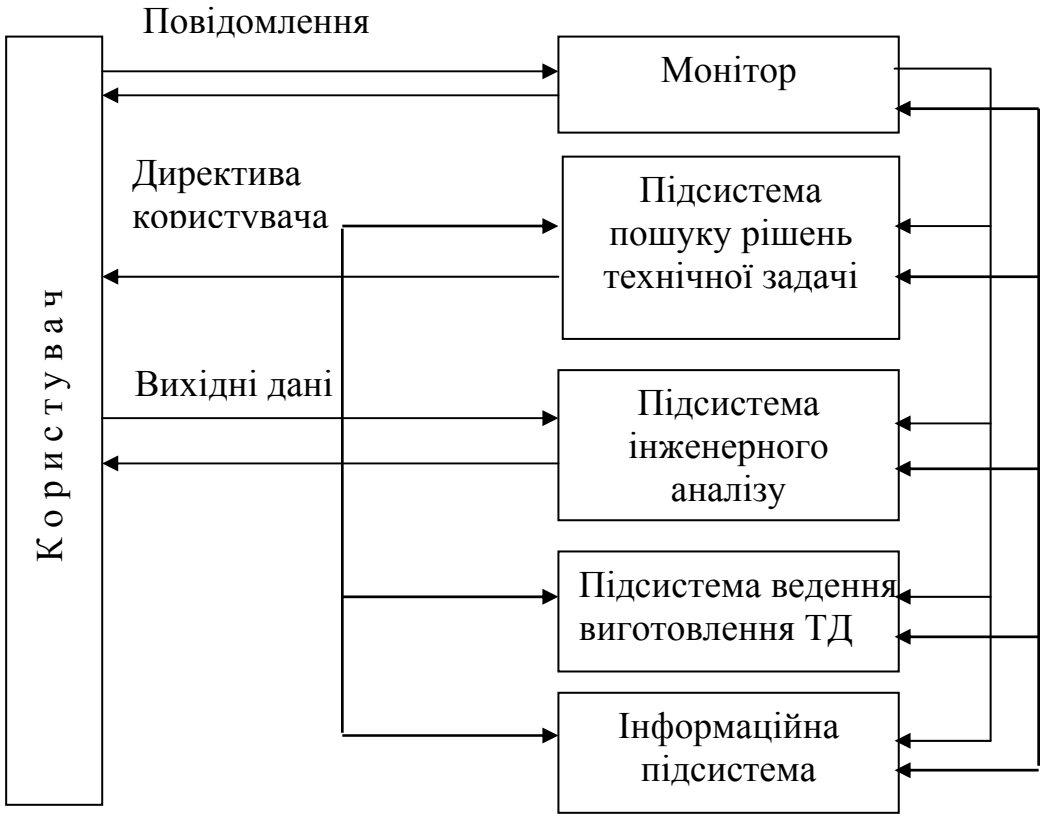


Рисунок 3 – Базові підсистеми САПР

2 Підсистеми САПР

2.1 Інформаційна підсистема

Основною задачею інформаційної підсистеми в діючій САПР є забезпечення виконання системою наступних функцій: збирання, збереження, упорядкування, пошук, поповнення, виведення за вимогою користувача всієї необхідної для забезпечення процесу проектування документації (інформації). Застосування ЕОМ дозволяє представити цю інформацію в якості бази даних - сукупності упорядкованих комплексних зведень про ОП [21], [2], що включають:

а) світовий науково-технічний рівень (у виді публікацій, описів, патентів, винаходів);

б) фонд методів генерації варіантів рішень, включаючи синтез з бібліотеками фізичних ефектів;

в) методики проектування , що представляють собою формалізований досвід колективів розробників у даній області пошуку проектних рішень;

г) опис параметрів і характеристик об’єктів проектування, їх моделей для різних етапів проектування;

д) архів накопичених проектних рішень (як усієї задачі в цілому, так і її окремих фрагментів);

е) опис типів елементів, матеріалів;

ж) керівні і довідкові матеріали, стандарти й інші документи, які регламентують процес проектування.

Інформаційну підсистему САПР обслуговує СУБД, яка входить до складу банку даних, і яка регулює механізм доступу користувача у залежності від його запитів і пріоритету.

Очевидно, що чим більшою кількістю інформації володіє САПР, тим більша кількість знань переходить в ОП, тим більше живої праці буде зекономлено в ході розробки, організації виробництва й експлуатації об’єкта проектування.

З цього погляду процес проектування - це безперервне споживання і переробка інформації, у результаті чого відбувається генерування і добір варіантів побудови проекту, їх аналіз і оптимізація.

Побудова баз даних (БД) - складний і трудомісткий процес, що визначає багато в чому ефективність

функціонування системи автоматизованого проектування. Крім того, БД повинна бути гнучкою, тобто вільно прив’язуватися до умов конкретного підприємства даної спеціалізації ОП [30], [22].

Формалізація предметної області САПР повинна бути зафіксована в словнику понять, який описує об’єкти проектування для конкретної САПР та визначає їх семантичне значення.

БД зі своєю системою керування утворять банк даних (БД).

Інформаційна підсистема поповнюється даними, з неї витягається інформація для ручних, інтерактивних, автоматизованих і автоматичних етапів проектування. Наявність у САПР інформаційної підсистеми дозволяє мати:

- повну й актуальну інформацію про об’єкт проектування;

- скоротити обсяг і підвищити вірогідність даних;

- використовувати засоби, які надаються системою керування для ведення БД, захисту від несанкціонованого доступу і т.д.

Важливою характеристикою САПР є ступінь її інформаційного зв’язку з навколишнім середовищем. Система називається статичною, якщо в процесі проектування не потрібна інформація про поточний стан зовнішнього середовища, і динамічною, якщо при функціонуванні система безупинно споживає інформацію про стан зовнішнього середовища.

Основними компонентами інформаційної підсистеми є наступні блоки:

- структури даних;
- структури пам’яті;
- системи кодування;
- системи обміну з користувачами;
- системи типів керування і характеру обробки даних.

Експлуатація підсистеми, її поповнення й удосконалення, постійно виходять за межі можливостей конкретного проектувальника, кваліфікованого фахівця у своїй області. Це пояснюється тим, що частину задач можуть вирішити тільки фахівці відповідного профілю, що входять в організаційну структуру САПР. Наприклад, можна рекомендувати для використання в САПР СУБД ІНЕС, розроблену у ВНДСІ; для технологічної підготовки виробництва систему АСПТД, розроблену Донецьким НІІКЕ; для трасування друкованих плат – PKAD; для проектно-конструкторських робіт - ІТЕКАН(АН Білорусії), Автокад і ін.

2.2 Підсистема пошуку рішення технічної задачі

На початковій стадії розробки ОП виникає потреба в пошуку і розробці нових технічних ідей або пошук існуючих аналогів у БД [2]. Умовно можна виділити 2 класи:

1) пошук нових принципів розробки технічних об’єктів;

2) пошук вже існуючих варіантів рішення (ВР).

Методик пошуку рішення технічних задач за допомогою ЕОМ багато, що робить цей пошук цілеспрямованим і ефективним і дозволяє виключити рутинні елементи процесу проектування [21], [25]. Ефективність застосування методів і автоматизованих систем генерації раціонального варіанта рішення технічної задачі можна оцінити наступними показниками:

- у десятки і сотні разів скорочується час розробки конструктивних рішень для створюваних ОП або модернізації рішень, які вже віднайдені в БД;

- генеруються рішення, близькі до глобально оптимальних, у результаті чого в середньому на 20...30 % підвищується якість проектованих виробів;

- методи автоматизованої генерації рішень підвищують інтелектуальні здібності і творчу активність проектувальника.

Розробка варіантів рішень технічної задачі відповідає творчому етапові проектування, при реалізації якого використовується всі знання й уміння конкретного фахівця, тому автоматизація цієї задачі представляє один з найважливіших напрямків у проблемі використання штучного інтелекту в САПР.

З огляду на важливість цієї задачі, в організації структури САПР бажано передбачити групу фахівців, які будуть поповнювати підсистему новими алгоритмами творчості, що враховують як особливості характеру і структури розв’язуваних задач, так і особливості психічної діяльності людини. З цього погляду вже зараз можна говорити про

налаштування САПР на конкретного фахівця, про сполучення його особистості з можливостями САПР.

2.3 Підсистема інженерного аналізу (моделювання об'єкта й оптимізація його характеристик)

Основне призначення підсистеми інженерного аналізу полягає в виконанні усіх обчислювальних робіт, пов'язаних з деталізацією виконаного проектного рішення [2]. Важливість такої базової підсистеми визначається загальною тенденцією математизації науки і техніки, при цьому, використання більш складних моделей і більш могутніх ЗОТ дозволяє значно наблизити показники моделі до дійсних параметрів об'єкта проектування.

Арсенал обчислювальних методів поповнюється постійно, багато знову виникаючих інженерних задач стимулюють розробку нових підходів і методів, нових критеріїв і алгоритмів.

Можливі різні види моделей, які використовуються для вибору в залежності від параметрів об'єкта:

- аналітичні (детерміновані - безперервні і дискретні, стохастичні), причому етапам ескізного, технічного, конструкторського і технологічного проектування відповідають свої моделі;

- імітаційні, якщо об'єкт відрізняється невизначеністю проектування. Такі моделі відтворюють процес функціонування ОП, а оцінка різних варіантів рішень при зміні керуючих перемінних дозволяє знайти найбільш оптимальний;

- евристичні або ігрові моделі, коли об’єкт характеризується невизначеністю функціонування і не встановлені значення його параметрів. У таких умовах використовується інтуїтивний вибір в умовах неповної інформації.

У процесі проектування моделі розглядаються в зворотному порядку: від загальних до самих точних.

Методика САПР постійно уточнюється й удосконалюється, тому структура САПР повинна доповнюватися зміною окремих частин при збереженні самої структури. Цим вимогам найбільше повно відповідають структури інтегрованих САПР. Такі системи характеризуються модульним принципом побудови програмного забезпечення, наявністю убудованих ОС і набором альтернативних проектних процедур.

Розробку й експлуатацію підсистем інженерного аналізу здійснюють спеціальні фахівці, що організовують структури САПР, однак у тісній співдружності з проектувальником - користувачем, тому що тільки він зможе уточнювати і формувати математичні моделі ОП, пропонувати модифікації методики проектування, виходячи з вихідних задач.

2.4 Підсистема ведення і виготовлення документації

Проектно-конструкторська документація - основний результат функціонування САПР [24], [25]. Це графічні, тестові документи, фотошаблони і т.д.

Призначення підсистеми - одержання проектних документів, необхідних для виготовлення ОП у виробництві. Розробка і виготовлення КД звичайне складає 45-60% усіх витрат часу на проектування ОП, причому це дійсно рутинна робота, одноманітна і стомлююча, не говорячи вже про наступне багаторазове корегування, яке завжди присутнє при наявності людського фактору в процесі проектування.

Автоматичне та автоматизоване проектування КД, що заміняє людину на цих операціях, здійснюється за допомогою наступних технічних засобів:

- КГА (креслярсько-графічні автомати);
- графопобудовників;
- пристроїв мікрофільмування;
- пристроїв репродукування.

Даною підсистемою здійснюється компонування документів, тобто розбивка їх на окремі формати. Використовувані для одержання документації пристрої, умовно можна розділити на 2 типи:

- швидкодіючі дисплеї й АЦПУ;
- повільні графопобудовники, які надають можливість одержати закінчену графічну документацію.

3 Призначення підсистем САПР

Розглянуті підсистеми - основа нової технології створення САПР і стосовно ОП є інваріантними. В умовах роботи САПР проектувальнику досить лише знати правила запису ТЗ на

проектування за допомогою спеціальних мов і директив керування системою, щоб ініціалізувати процес обробки і відображення інформації в САПР. Діалоговий режим дозволяє оперативно відслідковувати і корегувати процес проектування.

Підсистеми і компоненти САПР з'єднуються і взаємодіють один з одним під керуванням ОС, що відображає логічну схему побудови об'єкта проектування відповідно до директив користувача.

Попадаючи під вплив проектних процедур, модель проекту розвивається, накопичуючи інформацію таким чином, щоб у будь-який момент надати її конструктору. Така організація дозволяє досягти рішення найважливішої задачі - забезпечення єдності моделі проекту на всіх стадіях і фазах розробки, що і відрізняє САПР від набору роз'єднаних програм.

Системі АПР властиво:

- інформаційна модель будується в пам'яті ЕОМ при виконанні проектних процедур за допомогою ЕОМ;

- розвиток моделі відбувається в результаті безпосереднього її взаємодії з різними групами користувачів;

- за вказівкою користувача модель може бути піддана кожній із процедур, що знаходяться в бібліотеці системи.

КЛАСИФІКАЦІЯ САПР

1 Загальне положення класифікації САПР

2 Класифікація по етапах розвитку ЕОМ

3 Класифікація по класах використовуваних ЕОМ

4 Класифікація по можливостях, пропонованих користувачам САПР

5 Класифікація по маршрутах проектування ВЕТ

5.1 ІНСАПР

5.2 ССАПР

5.3 САВПР

5.4 ГСАПР

6 Питання освоєння і подальшого розвитку САПР ВЕТ

1. Загальне положення класифікації САПР

У класифікації САПР можна виділити 4 напрямки [26]:

- класифікація по етапах розвитку ЕОМ;
- класифікація по класах використовуваних ЕОМ;
- класифікація по можливостях пропонованих ЕОМ;
- класифікація по маршрутах проектування ІЕТ.

2. Класифікація по етапах розвитку ЕОМ

Розрізняють 4 покоління САПР [26]:

Перше – це технічні засоби типу ЕОМ М – 220, БЕСМ – 4, РАЗДАН, МІНСЬК – 22. Приклади САПР: вузько направлені завдання «АВРОРА», «АСП – 1», «Автограф», «ПА – 1». На

сьогодення це покоління не використовується в проектних організаціях та на виробництві жодної розвиненої країни світу і є фактично вже тільки історичним етапом початку створення і подальшого розвитку САПР та інших комп'ютерних систем (як друге і третє покоління).

Друге – використовує технічні засоби ЕОМ БЕСМ – 4, Електроніка – 60, М – 6000, координатографи і графічні пристрої КПА – 1200, МІНСЬК 2004, 2005; ЕМ – 583, ЕМ – 209.

Третє – будувалося на основі ЕОМ серії ЄС із залученням міні-ЕОМ «Електроніка», СМ – 4, СМ – 1420.

Четверте – на основі сучасних ПК та комп'ютерних мереж.

3. Класифікація по класах розвитку ЕОМ

Перший клас – САПР на основі спеціалізованих робочих місць проектувальника та ПК. Призначення таких систем – автоматизація проектування фотошаблонів друкованих плат, мікросхем, управління верстатами з ЧПУ при виготовленні механічних конструкцій ВЕТ.

Другий клас – САПР на основі створених локальних комп'ютерних мереж, що збільшує потужність і надійність систем, комплексування ЕОМ дозволяє реалізувати ієрархічну структуру технічних засобів САПР (2.3 рівня).

Третій клас – САПР, в яких реалізовані системи розділення часу і режим мультидоступа на основі використання

комп'ютерних локальних та глобальних мереж. Це дало можливість провести в організації – розробника розділення роботи над проектом (одночасне проектування окремих частин). Ядро технічних засобів – супер - ЕОМ.

Четвертий клас – характеризується наявністю ЕОМ всіх рівнів і мереж. Такі мережі будуються як на рівні підприємства, так і на рівні галузі; широко використовуються глобальні мережі. Мережева структура дозволяє в багато разів збільшувати потужність САПР, створювати інтегровані банки даних.

4. Класифікація по можливостях, пропонованих користувачам САПР

Розрізняють інженерні, спеціалізовані, універсальні і унікальні САПР.

Інженерні САПР – реалізовані на ПК. Призначені для виконання окремих видів інженерних розрахунків і проектних робіт. Деякі інженерні САПР називають АРМ (автоматизоване робоче місце) [7].

Приклад:

АРМ – З – АРМ схемотехніка;

АРМ – Т – АРМ технолога;

АРМ – ТП – АРМ тополога.

Інженерні САПР відповідають САПР 1-го класу.

Спеціалізовані САПР – системи колективного користування, орієнтовані на виконання найбільш масових проектних завдань. У основі: використання математичних моделей, методів моделювання і оптимізації на всіх етапах

проектування. Спеціалізовані САПР відповідають другому класу.

Універсальні САПР – часто носять галузевий характер і забезпечують багатокористувацький режим при проектуванні всієї номенклатури виробів. Технічні засоби формуються по 2-х або 3-х рівневому ієрархічному принципу. На 1-му рівні – супер-ЕОМ; на 2-му – середні ЕОМ і міні-ЕОМ, обслуговуючі окремі термінали; на третьому - міні-, мікро-ЕоМ, що являють собою окремі робочі місця. Універсальні САПР відповідають третьому класу.

Унікальні САПР – носять міжгалузевий характер і створюються для вирішення крупних народногосподарських завдань, вони відповідають 4-у класу.

5. Класифікація по маршрутах проектування ОП

Розрізняють [18], [19]:

- ІНСАП;
- ССАПР;
- САВПР;
- ГСАПР.

Інтегровані САПР (ІНСАПР) – характеризуються широким спектром вирішення завдань на будь-якому рівні. Технічні засоби – найчастіше ЕОМ з централізованим обчислювальним комплексом на верхньому рівні ієрархії і ПК на нижньому рівні ієрархії. ІНСАПР особливо ефективні при розробці нових ВЕТ з урахуванням нових технологій і матеріалів [1].

Наскрізні САПР (ССАПР) – вирішення завдань за конвеєрним принципом обробки інформації. Технічні засоби – ЕОМ середнього класу і швидкодії. Ефективність в основному залежить від якості розробки системного і програмного забезпечення. Переналаштування на проектування іншого типу ОП – є трудомісткою та економічно недоцільною задачею. Окупність можлива тільки при виробництві великої партії ВЕТ одного типу [27].

Система автоматичного проектування (САВПР) – знайшли основне застосування при розробці елементної бази ВЕТ. Реалізується на будь-якому класі ЕОМ. Характерна риса – мінімальна участь людини в процесі проектування. САВПР – це подальший розвиток САПР, за принципом побудови вони близькі до систем з штучним інтелектом.

Гнучкі САПР (ГСАПР) – розвиток отримали паралельно з гнучкими автоматизованими виробництвами. Характеризуються властивістю гнучкого налаштування системи на необхідні технічні засоби і клас рішення задачі. Перша властивість називається мобільністю, друга – адаптивністю.

Приведена класифікація володіє взаємною відповідністю і має наступне графічне представлення, показане відповідно до рисунку 1.

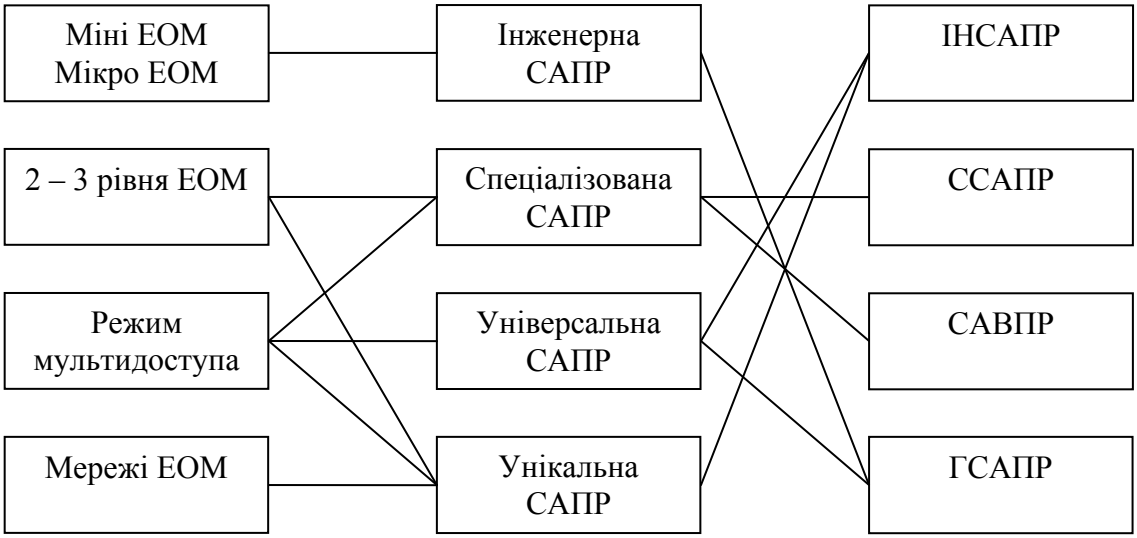


Рисунок 1 – Схема взаємної відповідності класів САПР

САПР – складна людино-машинна система, яка є складовою підкласу інформаційних кібернетичних систем. Взаємодію основних складових САПР можна представити відповідно до рисунка 2 схемою, де структурно відображена взаємодія користувача, ЕОМ, ППЗ, ОС.

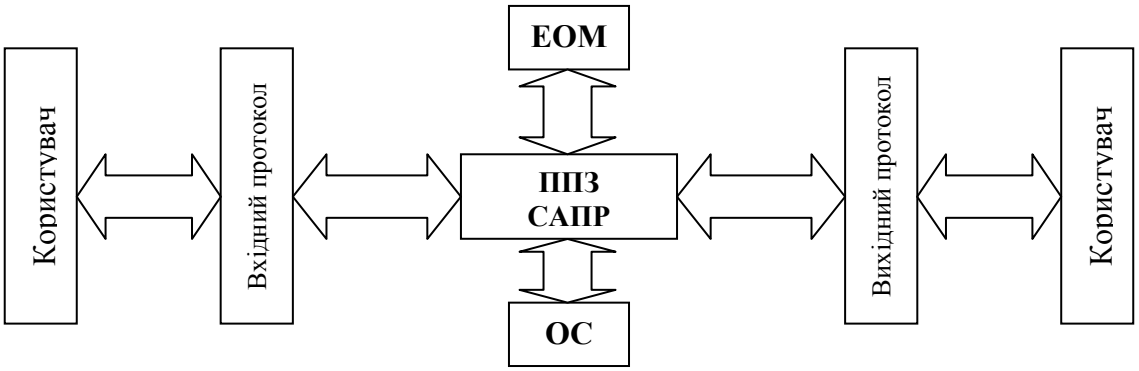


Рисунок 2 – Структурна схема взаємодії основних складових САПР

Вхідний протокол – спілкування користувача з ПК на будь-якій машинній мові з метою завдання необхідного режиму роботи і опису ОП.

Вихідний протокол – результати роботи ПК по вхідному протоколу у вигляді графічної і алфавітно-цифрової інформації.

5.1 ІНСАПР

Найпоширеніші системи проектування ВЕТ і процесу їх виготовлення. Проблеми їх створення найбільш пропрацювали, структура повністю склалася.

Розглянемо загальні питання побудови ІНСАПР. Програмне забезпечення містить:

- ПРОПМ – проблемно-орієнтований програмний модуль;
- УП – управляючі програми;
- БНД – банки даних.

Типова структура ПРОПМ надана відповідно до рис. 3.

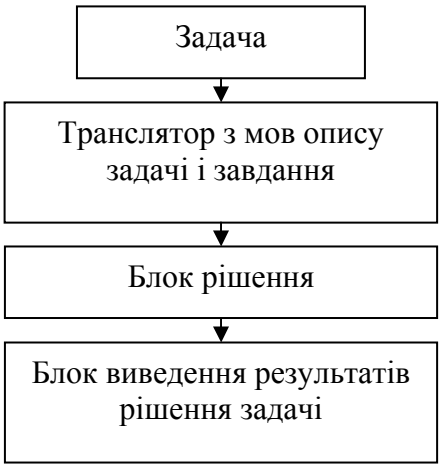


Рисунок 3 – Структурна схема ПРОПМ

Даний модуль дозволяє вирішувати окремі прикладні завдання. Він містить:

- вхідну частину, оброблювальну мову опису технічного завдання;

- блок рішення, оброблювальна ММ;

- вихідну частину, оброблювальну передавальну або частину виведення результатів блоку рішень.

ПРОПМ об’єднуються (інтегруються) в єдину САПР із структурою, показаною на рисунку 4.

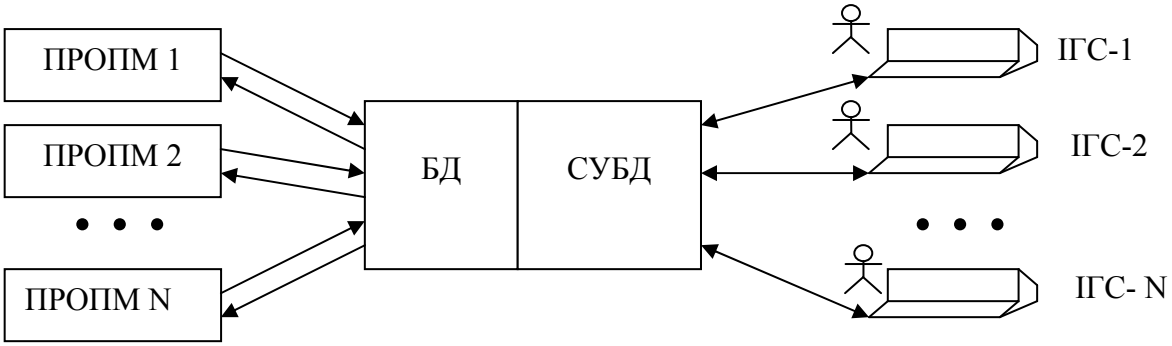


Рисунок 4 – Структурна схема інтеграції ПРОПМ в єдину САПР: ІГС – інтерактивно-графічна станція.

Для однієї і тієї ж ІНСАПР, ПРОПМ можуть створювати різні організації, при цьому при їх інтеграції в єдину систему, вхідні і вихідні протоколи зберігаються. У процесі роботи з системою користувач викликає через ІГС відповідний вирішуваному завданню ПРОПМ з банку даних. Результати роботи, згідно внутрішньому уніфікованому представленню даних, заносяться в БД і можуть бути неодноразово використані (навіть з іншими ПРОПМ).

Окремі ПРОПМ можна інтегрувати і в ППП цільового призначення (але зі своїми мовами і сумісністю в межах даного ППП).

ІНСАПР призначені для створення і дослідження нових системно-схемно-технологічних рішень на підприємствах різного призначення. Вони ж інтегрують досвід програмістів – розробників за багато років.

Такі системи вимагають дуже розвинених обчислювальних засобів, зразкову структуру яких можна представити схемою відповідно до рисунку 5.

Розглянемо деталізовану структуру ІНСАПР (рисунок 6).

Основа ІНСАПР – єдина база даних. Удосконалюється ІНСАПР по шляху поліпшення СУБД, єдиної БД, об’єднання мов опису ОП, спрощення операції додавання нового модуля або нових ММ ОП.

Основні вимоги до ІНСАПР:

- забезпечення простого доступу до ПРОПМ, що входять до складу системи;

- забезпечення прямого доступу користувача до обчислювальних засобів. Користувач повинен мати можливість:

- ставити завдання;

- вводити дані;

- отримувати результати проектних і розрахункових операцій.

- максимальна універсальність ППП;

- максимальна адаптація до умов проектування.

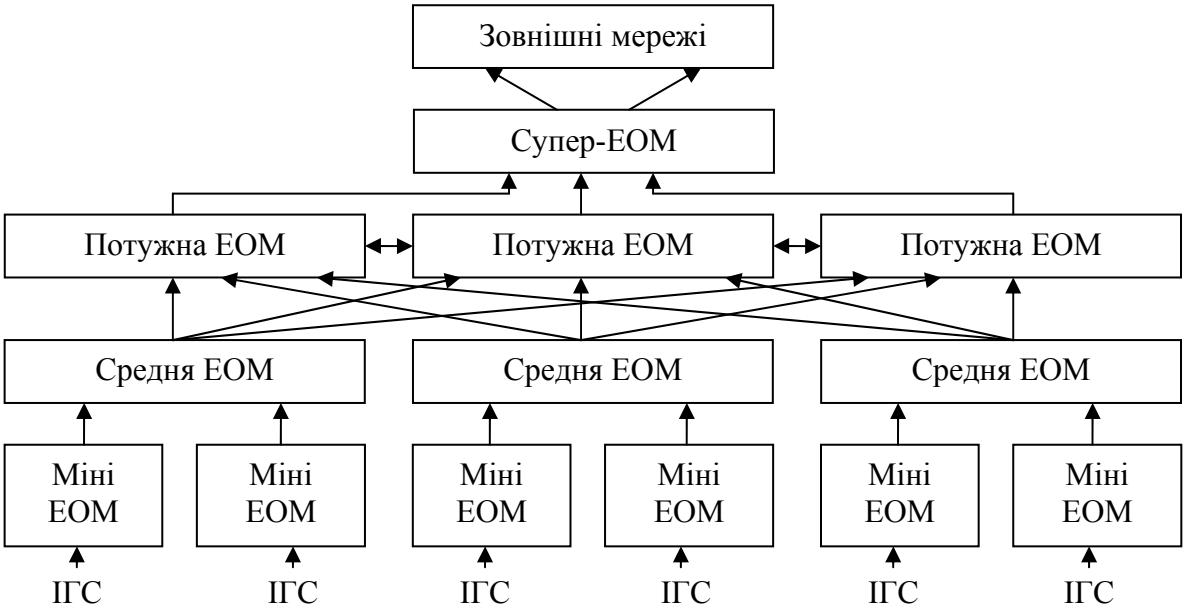


Рисунок 5 – Структурна схема інтеграції ЗОТ в ІНСАПР

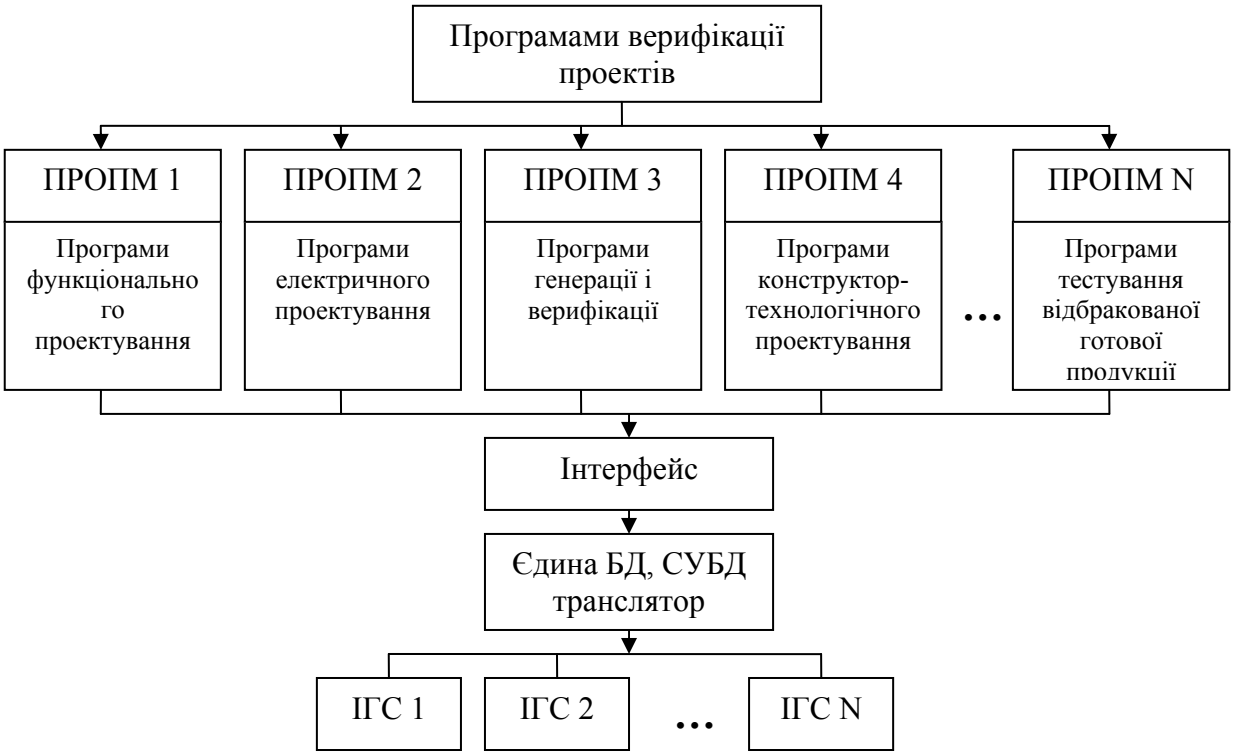


Рисунок 6 – Структурна схема деталізованої структури ІНСАПР

Умови проектування – це доповнення наявної в системі БД, корегування ПРОПМ. У основі реалізації – модульний

принцип побудови ПЗ і створення БД, пов’язаної інтерфейсом з ПРОПМ, забезпечення прямого виходу на виробничі автомати з числовим програмним управлінням.

Програмно – інформаційне забезпечення ІНСАПР включає:

- лінгвістичне;
- системне;
- програмне;
- прикладне програмне забезпечення

і має наступну структуру відповідно до рисунку 7.

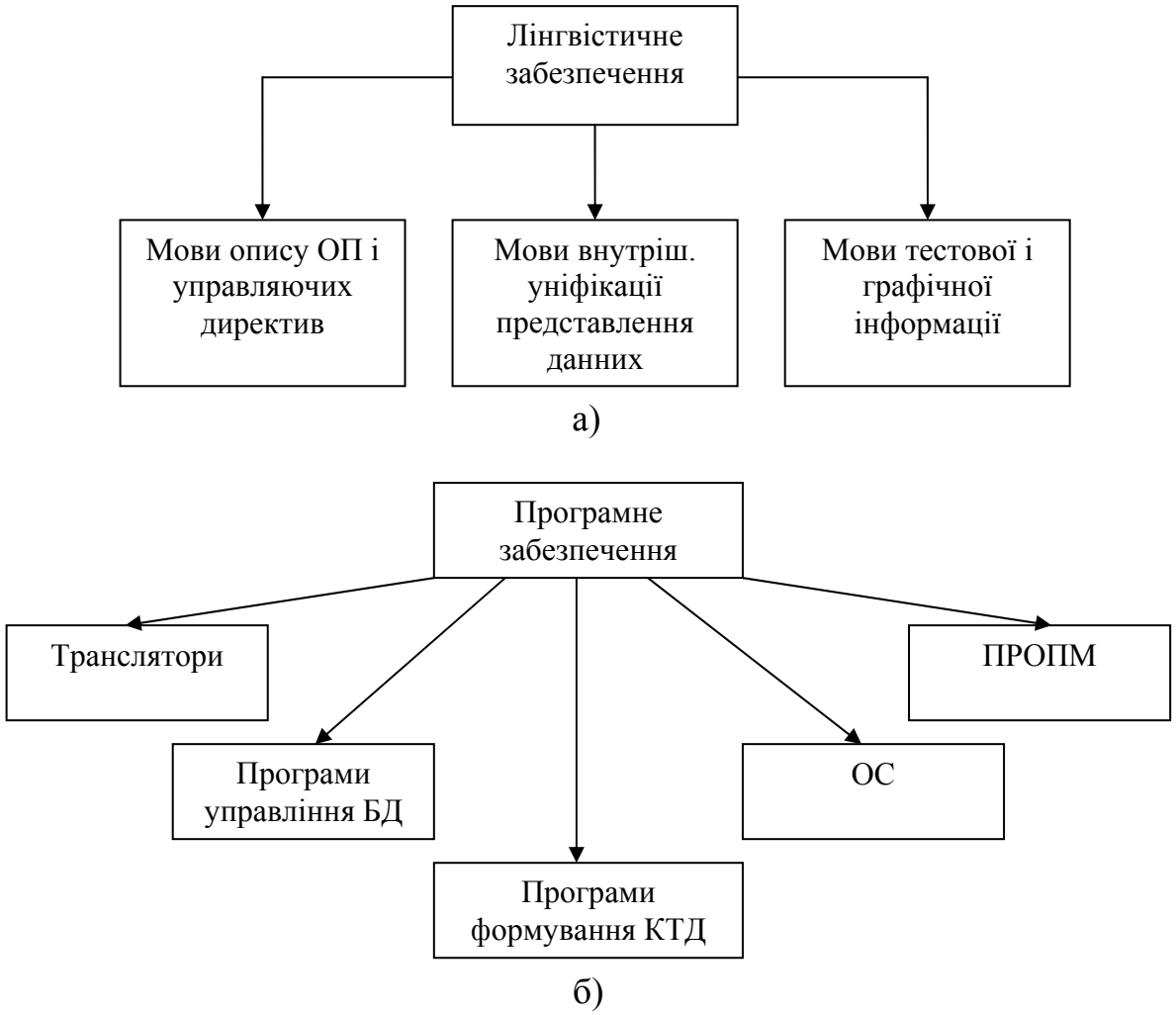


Рисунок 7 – Структурна схема лінгвістичного (а) і програмного забезпечення (б) ІНСАПР

ІНСАПР в основному орієнтовані на функціональне і конструкторсько-технологічне проектування ВЕТ різного призначення практично всього спектру. Спектри ВЕТ – це класифікація виробів РЕА по певних принципах, як це показано відповідно до рисунку 8.

Введення специфікації по спектрах дозволяє при розробці конкретної ІНСАПР вести чітку структуру майбутньої системи вже на початкових стадіях її розробки.

Характерною особливістю ІНСАПР є архітектура технічних засобів (одно- і дворівнева структура), як це показано на рисунку 9.

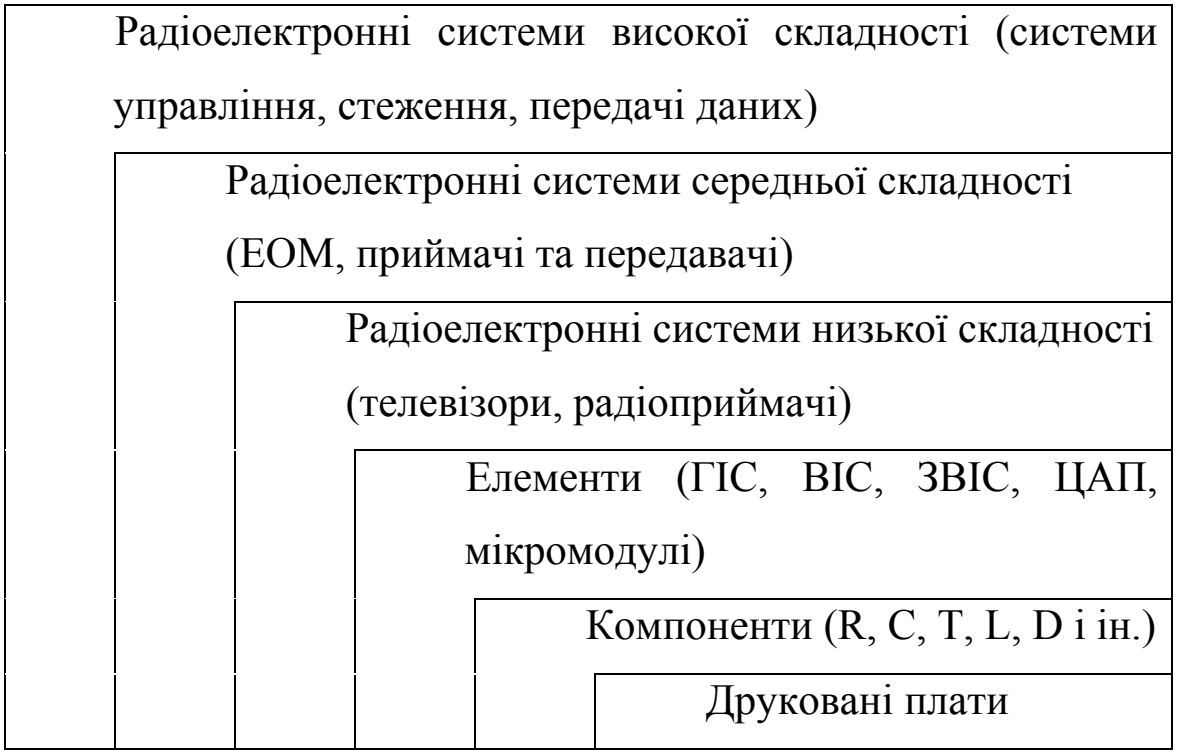


Рисунок 8 – Спектри ВЕТ

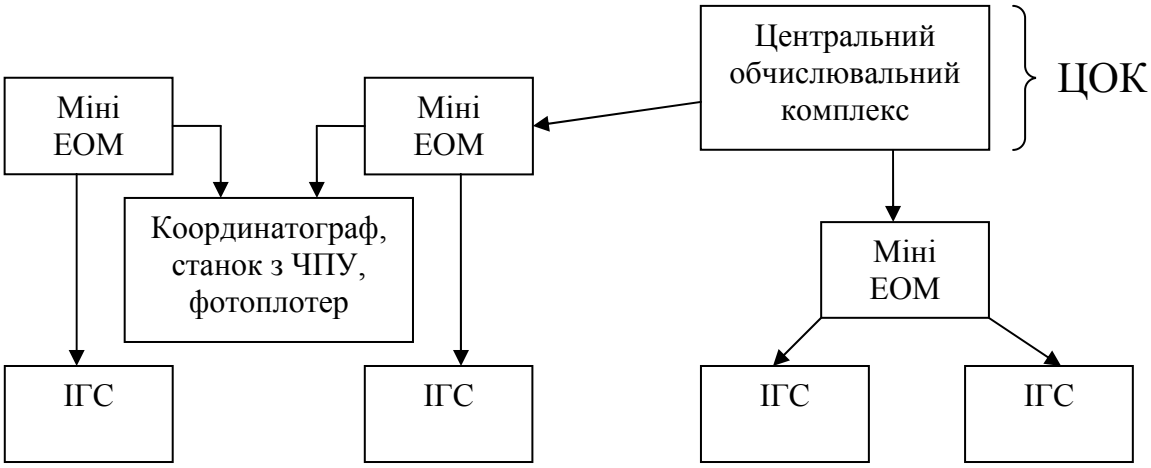


Рисунок 9 – Структурна схема архітектури технічних засобів САПР

Маршрут проходження проекту включає наступні етапи:

- введення початкових даних і їх коректування;
- проектування топології;
- доопрацювання проекту в інтерактивному режимі;
- отримання КТД.

Реалізація маршруту може відрізнятися залежно від розподілу функцій і способів взаємодії користувача і ІНСАПР. В даний час виділено і використовується 3 маршрути проектування ОП. Розглянемо детальніше кожен з них.

1-й маршрут

Введення підготовленого ескізу топології за допомогою координатографа (пристрої введення графічної інформації), сканера і далі створюється топологія, як це показано на структурній схемі відповідно до рисунку 10.

Рисунок 10 – Структурна схема маршруту проектування ОП №1

Тобто - маршрут на рівні підготовленого ескізу.

2-й маршрут

Введення даних в інтерактивному режимі за допомогою координатографа. Маршрут проектування той же, що і в першому випадку.

3-й маршрут

Вводяться тільки початкові дані (технічні вимоги до друкарської плати і схема електрична принципова). Проектування здійснюється в автоматичному режимі. Маршрут проектування матиме вигляд відповідно до рисунку 11.

В ІНСАПР автоматичний режим ніколи не реалізовується повністю. Найчастіше проектування проводиться в інтерактивному режимі. Ці системи і отримали найбільше застосування. У такій системі об’єднані в тісному контакті неодмінно високий інтелект користувача і ІНСАПР. Це дозволяє за короткий відрізок часу виконувати великий об’єм рутинної роботи під контролем користувача.

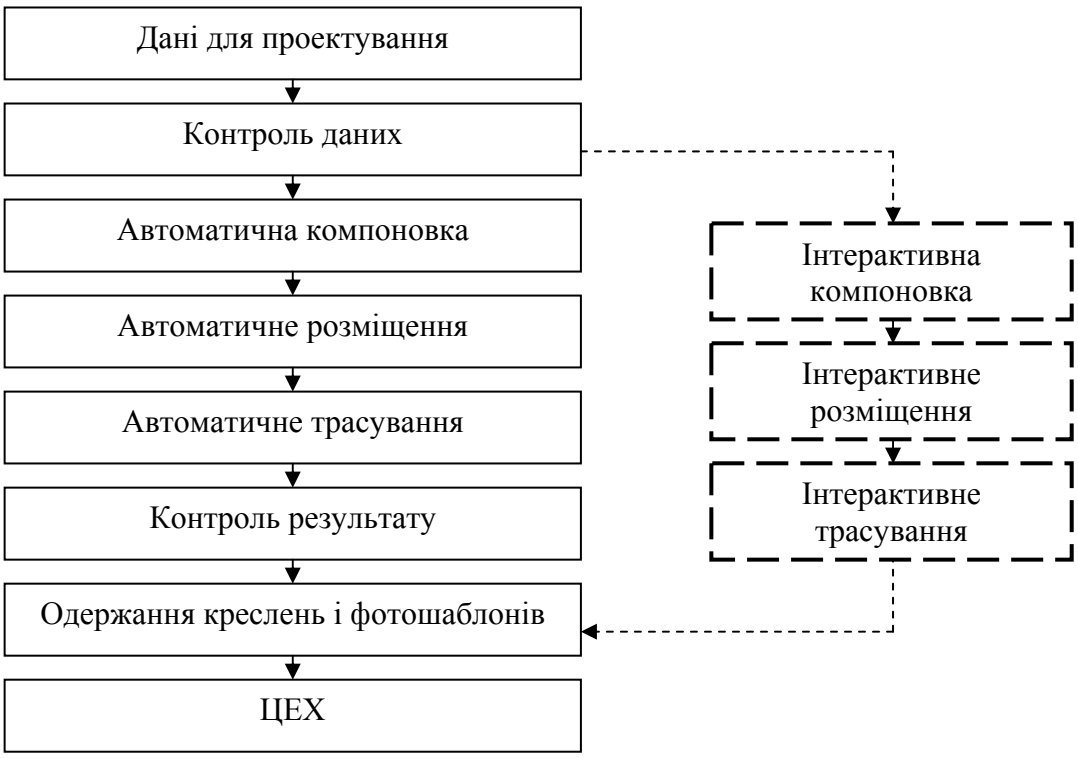


Рисунок 11 – Структурна схема маршруту проектування ОП №3

Наприклад, після введення даних, користувач отримує результати проектування, і в інтерактивному режимі може зробити виправлення одержаного результату і так далі

Таким чином, можна сформувати основну вимогу до інтерактивних систем – забезпечення на будь-якому етапі проектування можливості контролю і втручання людини.

В цілому ІНСАПР друкованих плат повинна реалізовувати наступні функції:

- видавати всі повідомлення про помилки в зручній для користувача формі;

- пропонувати користувачу інформацію по виправленню помилок;

- дозволяти розміщувати ЕРЕ на ДМП в інтерактивному режимі;

- зображати сигнальні ланцюги, пов'язані з деякими або всіма розміщуваними елементами;

- гарантувати відповідність НСІ, КТД і ТД на предмет забезпечення ЄСКД, ЄСТД та діючих ДСТУ та ISO;

- забезпечувати контроль і корегування проектних процедур трасування;

- забезпечити перехід від одного алгоритму проектування до іншого в автоматичному режимі.

Всі ці вимоги повною мірою реалізуються за допомогою ІГС, а простіше – комплексу апаратно-графічних засобів, що забезпечує взаємодію користувача з ЕОМ (термінальний пункт).

5.2 Системи наскрізного автоматизованого проектування (ССАПР)

ССАПР не завжди можна побудувати через нестачу потужних і розгалужених технічних засобів. Проекти не завжди оптимальні, а терміни проектування не задовольняють виробництво. Це і послужило поштовхом до створення ССАПР.

Характерні особливості:

- використання 1-о і 2-х рівневої архітектури технічних засобів;

- єдина мова опису ОП;

- єдина мова опису завдань;

- конвеєрна обробка інформації.

Найбільш широке застосування: проектування аналогових і цифрових схем малого, середнього, великого і надвеликого ступеня інтеграції.

Маршрут проходження проекту і склад математичного забезпечення ССАПР можна представити наступною структурною схемою відповідно до рисунку 12.

Після отримання ТЗ, користувач в інтерактивному режимі реалізує проект зверху вниз до виведення КТД і ТД. Інформація після кожного етапу видається на вимогу користувача на ІГС для проведення ним контролю. Інформація передається тільки від попереднього етапу до подальшого.

Особливість ССАПР – відсутність реальних баз даних, а також можливість альтернативного вибору того або іншого маршруту чи методу вирішення завдань: кожен попередній ППП інформаційно пов’язаний тільки з сусіднім.

ССАПР є системами, розробленими для проектування тільки однотипних виробів. Їх складно або не можливо взагалі переналаштувати на проектування інших елементів із-за необхідності зміни проблемної орієнтації прикладних програмних модулів. А чому? Відповідь проста – відсутня єдина БД, а значить інформаційне забезпечення кожний модуль має власне.

До переваг цих систем слід віднести наступне: завдяки послідовному проходженню проекту вірогідність появи помилки більш низька, ніж в ІНСАПР. На даний час ССАПР – це проміжна ланка на шляху побудови САВПР.

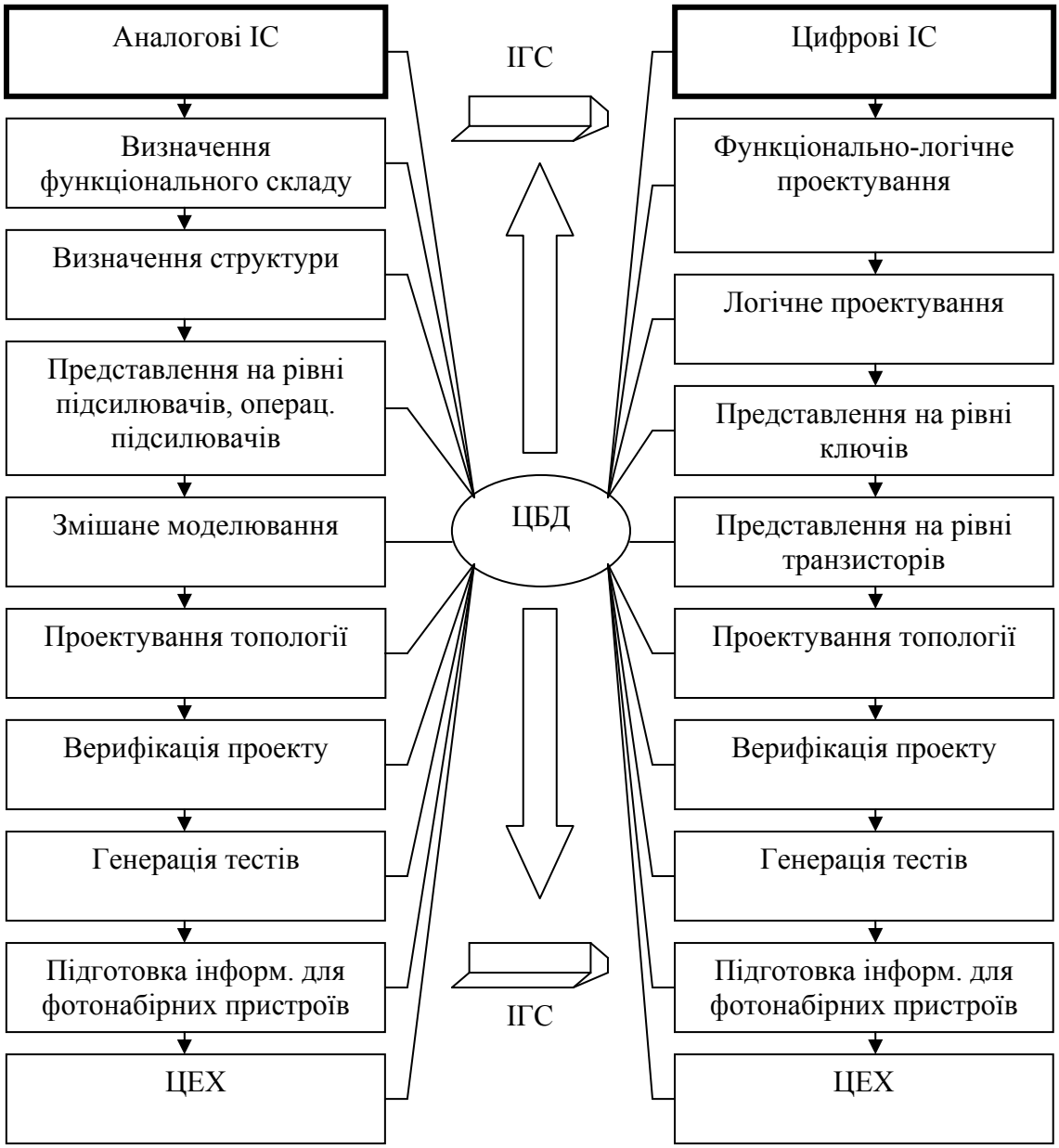


Рисунок 12 – Структурна схема ССАПР

Системи автоматичного проектування (САВПР)

Структурна схема САВПР має наступний вигляд (рисунок 13).

На етапі системного проектування визначається функціонування ОП, призначення і структура вхідних об’єктів, формується приватне ТЗ для проектування наступних етапів.

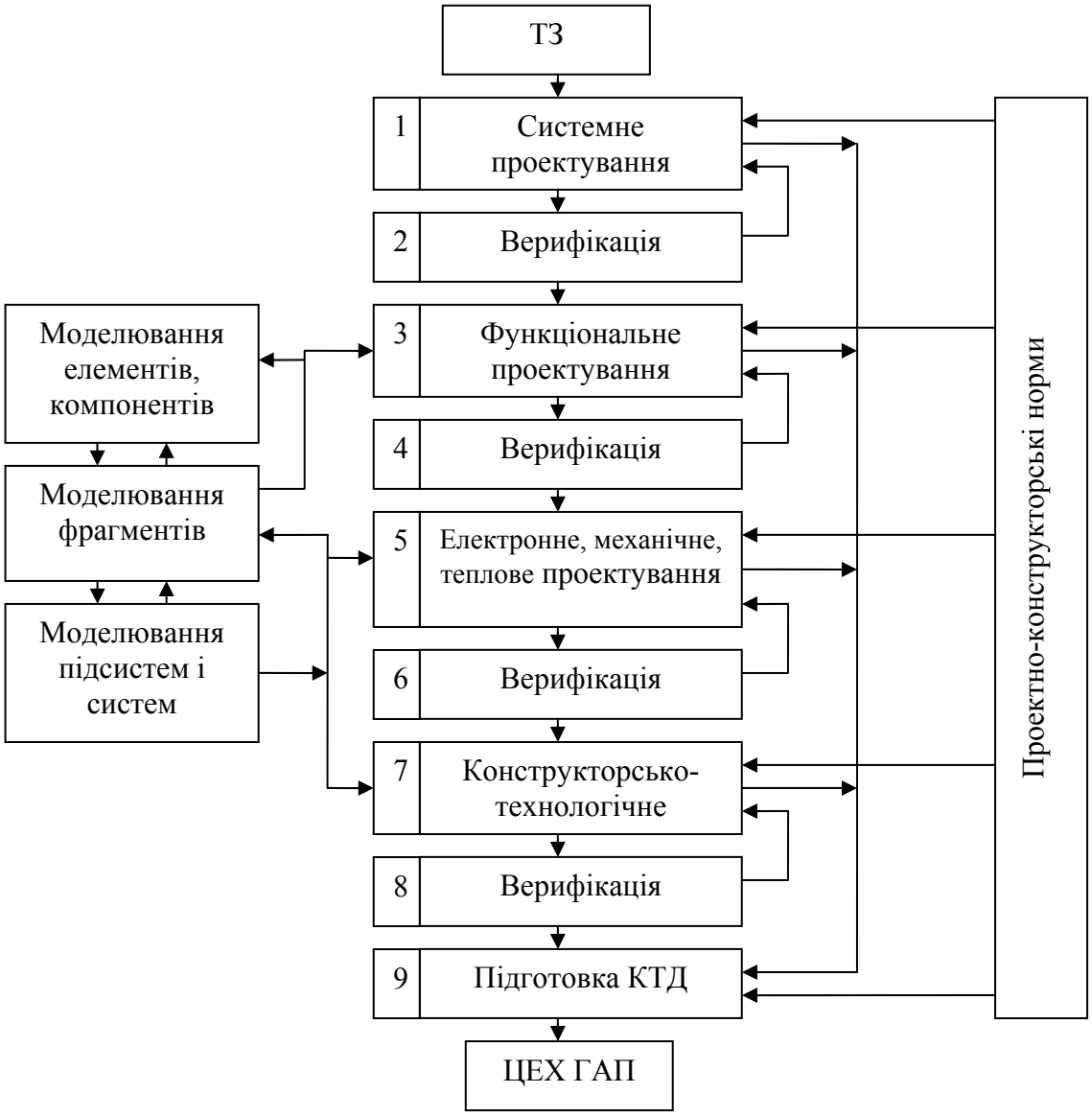


Рисунок 13 – Структурна схема САВПР

Далі проект уточнюється з урахуванням вимог системного проектування.

Конструкторсько-технологічне проектування дозволяє повністю визначити конструкцію ОП, монтажну схему і топологію.

Верифікація проекту – перевірка параметрів.

Вирішення показаних на маршруті завдань відбувається шляхом створення САПР, побудованих на основі використання наступних ідей:

- глобального дослідження простору проектних рішень;
- інтерактивних засобів проектування на основі дослідження локального простору проектних рішень;
- синтезу логічних матриць на основі базової технології;
- кремнієвих компіляторів (КРЕМКОМ);
- систем моделювання процесів творчої діяльності людини на основі штучного інтелекту.

Сучасні САПР ВЕТ і РЕА розвиваються у напрямі все більшої інтелектуалізації і, кінець кінцем, стануть повністю автоматичними.

Сучасний рівень розвитку електронної техніки дозволяє реалізувати перехід САПР до САВПР. Ідеальна модель САВПР наведена на рисунку 14.

Рисунок 14 – Ідеальна модель САВПР

Але це поки мета найближчого майбутнього. В даний час робляться наробки створення САВПР для вирішення завдань різних етапів проектування ЗВІС. Найбільш яскравим представником САВПР є КРЕМКОМ – кремнієвий (арсенідгалієвий) компілятор.

Структура САВПР аналогічна структурі ССАПР. Відмінність полягає в тому, що участь користувача необхідна тільки на етапі введення ТЗ і у тому випадку, коли САВПР не здатна вирішити поставлену проблему. У останньому випадку завдання вирішується в інтерактивному режимі роботи або за допомогою ІНСАПР або ССАПР.

Розберемо докладніше структуру КРЕМКОМ. Такі структури найбільш близькі до систем штучного інтелекту. Кінцевим продуктом проектування ЗВІС служить топологія, тобто інформація для виготовлення фотошаблону, що містить мільйони координат крапок.

Проект має ряд обмежень, звичайних при процедурі проектування. Тому для високого рівня кремнієвої мови необхідно створення кремнієвого компілятора для безпомилкової трансляції вхідної мови ТЗ в безпомилкову топологію.

Працюючі компілятори відтворюють топологію будь-якої складності за годину, тоді як на підготовку опису поведінки схеми потрібний, як мінімум тиждень.

Ще одна важлива особливість КРЕМКОМа, він працює за принципом «робити відразу добре» у відмінності від САПР - «робити і перевіряти».

Основа КРЕМКОМа – 2 мови:

- початкова – мова опису поведінки проектованої системи;

- цільова – мова опису можливостей кремнієвих технологій.

Робота з цими мовами можлива тільки за наявності розвиненого банку даних.

Якість КРЕМКОМа характеризується ступеню складності опису проведення системи, використовуваною площею на кристал і робочою частотою готової системи на кристалі.

Склад КРЕМКОМа:

- мова опису поведінки проекту;

- транслятор з початкової мови;

- блок моделювання системи;

- компілятор початкової мови в топологічному зображенні;

- інформація про фотошаблони для мови.

Основний компонент проектованої на кристалі системи – осередок з простими геометричними елементами. Прості мови розробляються за допомогою ІНСАПР і ССАПР і заносяться в банк даних. Кожен такий осередок є програмою, яка прорисувала відповідний елемент топології, трасує його, обчислює споживану потужність. Осередки з’єднуються між собою за допомогою контактних виводів.

Внутрішні деталі осередків і загальне розташування осередків залишаються невизначеними до завершальних етапів розробки, що дозволяє досягти максимальної гнучкості при проектуванні.

Користувач задає інформацію:

- формат мікроінструкцій;

- довжину слова;

- розрядність даних;

- специфікацію сигнальних шин;

- перелік елементів ядра ЗВІС;

- значення необхідних параметрів.

КРЕМКОМ синтезує топологію ядра, потім генерує частину, що управляє, ЗВІС.

Методологія проектування ЗВІС включає 4 етапи:

1) Опис майбутньої схеми – створення мікропрограм на проміжному рівні інтеграції на алгоритмічному підрівні, перетворення алгоритмів (множення замінюється послідовністю складань); трансляція інструкцій в заданий формат;

2) Перетворення операційної частини – генерація топології. Операційна частина організована у вигляді біт-модулів, архітектура і схеми електричні принципові стандартизовані. Це дозволяє вести їх проектування простою збіркою раніше розрахованих осередків. Біт-модуль – це елементарний арифметико-логічний пристрій (АЛУ);

3) Проектування частини, що управляє, – вибір проекту на основі ПЗП (програмованого запам’ятовуючого пристрою) або на основі програмованої логічної матриці з подальшою генерацією регулярних структур на основі специфікації;

4) Проектування проміжних частин – проектування блоків сполучення, блоків перетворення сигналів, вхідних і вихідних схем.

Для збереження надійності проекту, генератори операційних частин, що управляють, видають інформацію на

електричному і логічному рівнях на предмет визначення і усунення ризиків збоїв.

Структурна схема КРЕМКОМа.

Мова опису поведінки схеми – розробник описує ОП, задаючи тимчасові характеристики, порядок з'єднання входів і виходів елементів.

Екстрактор – будує схеми електричні принципові по топології інтегральних схем.

Генератор операційної частини – це синтезуючий блок обробки інформації на основі бібліотечних елементів.

Блок топологічного контролю – перевірка топології на виконання вимог КТД.

Генератор частини, що управляє, складається з підсистем:

- блок підсистем проекту, призначений для аналізу існуючих варіантів і вибору оптимального варіанту проекту;

- генератора перевіряючих частин, призначеного для тиражування на площині кристалу схем;

- оптимізатор, що проектує блоки вибірки із записом управляючої інформації.

Основним критерієм успішного функціонування САВПР є використання надійних ЗОТ.

Гнучкі САПР

ГСАПР – це часто умовний клас САПР. У клас ГСАПР, як правило, входять ІНСАПР і ССАПР.

Розробка 1 одиниці САПР, в середньому, триває від 3-х до 10 років. За цей період технічні засоби, як правило, міняються двічі. А значить періодично виникає проблема переведення

САПР з одних технічних засобів на інші, а якщо врахувати, що вартість розробки однієї системи 500 тис. – 1 млн.$ США, то економічні передумови створення мобільних САПР очевидні.

За рахунок чого можна вирішити проблему гнучкості?

Це:

- доопрацювання програмного забезпечення або заміна окремих ПРОПМ. Але цей варіант економічно не рентабельний, оскільки доопрацювання коштують не дешево;

- на стадії розробки програмного забезпечення передбачити можливість мобільності і адаптивності.

Шляхи вирішення завдань адаптації:

- перерозподіл функцій проектування між користувачем і САПР (якщо система не вирішує проблему, передати її на вирішення користувачу);

- включення альтернативних програм;

- створення банку моделей для розширення класу вирішуваних завдань;

- розробка алгоритмічних процедур, вільних від особливостей ОП (тобто, всі конкретні відомості виражені через параметри моделей);

- забезпечення алгоритмічної спадкоємності (унікальні алгоритми використовувати в нових САПР);

- безперервне вдосконалення теорії побудови САПР і методів програмування.

Шляхи вирішення завдань мобільності: при впровадженні і тиражуванні САПР виникає завдання перенесення програмного забезпечення на різнотипні ЕОМ. Тобто, мобільне

програмне забезпечення повинне забезпечувати використання різних типів ЕОМ без додаткової його розробки та трансляції.

Повністю мобільного програмного забезпечення поки не існує, але при його побудові можна і потрібно врахувати наступні принципи:

- застосування покрокової деталізації при написанні текстів програм;

- створення програмного забезпечення з динамічним розподілом ресурсів пам’яті;

- використання для написання програмного забезпечення мов високого рівня;

- застосування макросів – засобів для заміни однієї послідовності символів іншою.

Програмне забезпечення ГСАПР складається з трьох складових:

- варіантна частина (або залежна). Це програмне забезпечення, що стосується зв’язку з операційним середовищем, з підтримкою інтерактивного режиму проектування в системі;

- інваріантна частина до технічних засобів – це програми управління ходом проектування, обслуговування банків даних, трансляції мов опису ОП;

- інваріантна до прикладних завдань частина програмного забезпечення, яка забезпечує виконання системою наступних функцій: управління ходом проектування; вирішення ММ; вирішення систем рівнянь; проведення параметричного синтезу.

Структурна схема програмного забезпечення має наступний вигляд відповідно до рисунку 15.

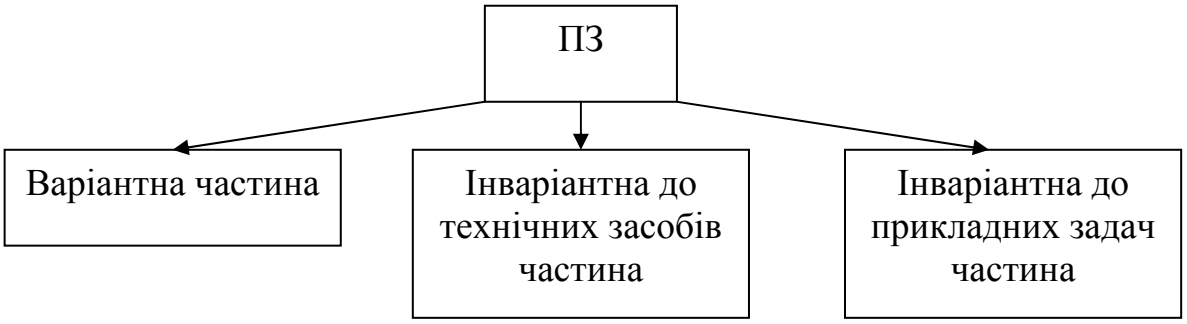


Рисунок 15 – Структурна схема ПЗ ГСАПР

6 Питання освоєння і подальшого розвитку САПР ВЕТ (ЗОТ)

САПР використовується в якості інструменту проектування, для максимальної ефективності використання украй важливими питаннями є підтримка її працездатності і впровадження.

Тому при навчанні, користувачу САПР необхідно дотримуватися певної методики. Зрозуміло, до навчання більш сприйнятливіший молодий користувач, а найбільший успіх в навчанні користувача можливий лише при контакті його з ЕОМ не менше 20 % часу навчання.

Методика навчання розробляється конструкторами системи і має три стандартні фази.

Перша фаза – детальне ознайомлення з правилами підготовки початкових даних і директивами управління ходом проектування. Контакт ЕОМ на цій фазі проводиться в кінці

періоду ознайомлення для закріплення навичок підготовки вхідної інформації.

Друга фаза – практична робота з ЕОМ з використанням завдань зростаючої трудності. На цьому етапі навчання обов'язкова допомога користувача, що вже освоїв дану САПР або конструктора САПР.

Третя фаза – практична робота з ЕОМ, освоєння загальної стратегії проектування і всього маршруту проектування, реалізованого в даній САПР.

Для успішного проведення більшості проектних процедур з допомогою САПР достатньо середньотехнічної освіти і знання основ інформатики і програмування.

Розглянемо шляхи розвитку трьох складових САПР:

- технічних засобів;
- системного програмного забезпечення;
- прикладного програмного забезпечення.

Технічні засоби.

Розвиток технічних засобів йде по трьох напрямках:

- створення могутніх надшвидкодіючих ЕОМ на основі багатопроцесорних систем. Такі ЕОМ дозволяють підтримувати автоматизоване проектування, моделювати складні моделі і процеси;

- створення СУПЕР-МІНІ-ЕОМ і ІГС. Вони повинні бути максимально наближені до користувача і містити елементи штучного інтелекту, щоб підсилити творчі здібності

проектувальника і сприяти зміні його соціального статусу в підході до інженерної справи;

- об’єднання ЕОМ цих двох класів в мережі локальні, міжгалузеві в масштабах загальноукраїнського значення. Наявність мереж зніме проблему впровадження САПР, перерозподілу обчислювальних ресурсів.

Системне програмне забезпечення.

Планується перерозподіл функцій САПР і ОС. Все більша кількість функцій управління і супроводу маршруту проходження проекту буде включаться в ОС. ОС, у свою чергу, спеціалізуватиметься тільки на обслуговування САПР або тільки на обслуговуванні завдань АСУ, АСНД або АСТПП. За допомогою спеціалізованих ОС можна створювати закриті САПР, тобто ОС по певному закону, виробленому замовником-користувачем, збиратиме потрібну САПР з наявних в БД модулів. Наприклад: САПР друкованих плат, САПР блоків ЕОМ і ін.

Прикладне програмне забезпечення.

ПЗ – апаратна реалізація ПРОПМ. При цьому, наприклад, програма рішення квадратного рівняння реалізується у вигляді електричної схеми або ж програма рішення системи алгебраїчних рівнянь реалізується у вигляді ВІС.

Розглядаючи різні класи САПР і тенденції їх розвитку, можна сформулювати вимоги до перспективної САПР:

- забезпечення автоматичної генерації інформації про проект;

- можливість використання мов високого рівня, близьких до природної мови спілкування;

- адаптація інтерактивних систем взаємодії до потреб користувача;

- можливість санкціонованого доступу;

- наявність автоматизованих банків даних;

- простота обслуговування системним фахівцем;

- швидкість налаштування САПР на необхідний об’єкт.

ЗАВДАННЯ СИНТЕЗУ Й АНАЛІЗУ

1 Загальні положення

2 Синтез і аналіз технічних рішень

2.1 Структурний синтез

2.2 Синтез припустимих технічних рішень

3. Методологія розв’язання задач структурного синтезу

3.1 Параметричний синтез

3.2 Математичне моделювання й параметричний синтез

3.3 Методи оптимізації в проектуванні технічних систем

4 Вибір раціональних варіантів рішення технічного завдання

4.1 Послідовний аналіз

4.2 Метод Парето

4.3 Метод гілок і меж.

1. Загальні положення

Як відомо, початковою стадією проектування є розробка, на підставі наявного ТЗ, технічної пропозиції й ескізного проекту, коли необхідно зробити вибір варіанту вузлів виробу або виробу в цілому.

На цих стадіях формується фізична й технічна основа майбутнього виробу, визначаються його техніко-економічні показники.

Отут і виникає невелика, здавалася б, проблема - наявність досвідченого відповідального проектувальника, що

володіє знаннями властивостей усього простору рішень. Тільки це може забезпечити на початковій стадії створення чіткого алгоритму, який би забезпечив цілеспрямований рух від формалізованого ТЗ убік поліпшеного технічного рішення, як це показано на рисунку 1. Ця процедура і визначає сутність прямого синтезу [21].

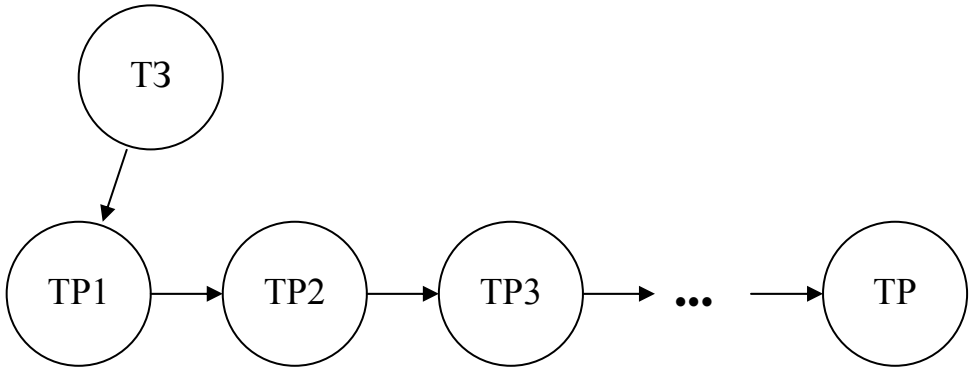


Рисунок 1 – Структурна схема прямого синтезу

Всі ви зіштовхувалися з вирішенням цього завдання на першому практичному занятті. Одержані в якості вихідного завдання на проектування друкованої плати в інтерактивному режимі фрагменти схеми електричної принципової нескладні та незначні по обсягу, однак варіантів компонування ЕРЕ виникло декілька, а що ж говорити про складні СХЕ, де їх - сотні й тисячі. Ви пішли в бібліотеку й спробували детально обстежити на рівні деталізації свій об’єкт проектування й переконалися , що це зайняло не так мало часу. А великий виріб? Чи можливо обстежити такий простір із прийнятним ступенем деталізації за час в межах розумного для людини? Ні.

Однак завдання генерації (синтезу) рішень можна вирішити й іншим способом, що виключає інтуїтивні рішення

фахівців. Існуючі САПР представляють розробнику при вирішенні самого відповідального етапу проектування - синтезу й аналізу топології друкованої плати - допомогу ЕОМ у виборі оптимального варіанту рішення.

У проектуванні виробів РЭА прийнято розрізняти два основних етапи синтезу: перший - вибір структурної схеми (структурний синтез) і другий - визначення параметрів обраної схеми по заданих властивостях (параметричний синтез).

2. Синтез і аналіз технічних рішень

2.1 Структурний синтез

Структурним синтезом називається процес створення структури ОП.

Структура ОП визначається природою елементів і способами їхнього зв’язку між собою в складі об’єкта проектування. Міжелементні зв’язки можуть бути функціональними, інформаційними, кінематичними, електричними, відображати взаємне розташування елементів, задавати впорядкування елементів у часі. Тому способи постановки задачі структурного синтезу для різних класів ОП досить різноманітні. Приміром, під структурним синтезом механізму розуміється проектування його схеми із зазначенням стійок рухливих ланок, видів кінематичних пар; при створенні електронних схем на етапі вибору структури потрібно визначити набір використовуваних стандартних елементів, зв’язки між ними, їхнє компонування та розташування в

дискретному монтажному просторі з урахуванням наявних обмежень.

Результатом структурного синтезу є структурна схема, представлена графічно із застосуванням умовних позначок або у вигляді переліку елементів з таблицею з'єднань, аналітичного запису , схеми алгоритму й т.д.

2.2 Синтез припустимих технічних рішень

Розроблена структурна схема повинна відповідати припустимому технічному рішенню, тобто задовольняти пред'явленому певному списку вимог - технічному завданню.

Зміст, порядок і узгодження ТЗ визначаються ДСТУ 22487-98. Воно повинне містити конкретні вимоги до вихідних параметрів, діапазони їхнього варіювання, вимоги й умови, описані на якісному рівні. ТЗ виділяє область припустимих значень параметрів ОП. Однак, для забезпечення вибору технічного рішення, близького до оптимального у відповідності з висунутими критеріями, у ТЗ повинні бути зазначені кращі значення вимог.

Попередній вибір фізичного принципу дії ОП на етапі узгодження ТЗ зменшує потужність безлічі припустимих типів елементів і їхніх зв'язків, тобто потужність безлічі можливих варіантів структури, і відповідно полегшує етап структурного синтезу при проектуванні. Однак, треба відзначити, що подібний підхід може призвести до втрати деякої кількості

допустимих рішень, серед яких можуть виявитись й близькі до оптимального.

3 Методологія рішення завдань структурного синтезу

Спроби автоматизувати етапи системотехнічного проектування призвели до появи оригінальних робіт, у яких викладаються різні підходи до рішення проблеми.

Всю розмаїтість методик пошуку технічних рішень можна класифікувати на:

- трансформаційні;

- формалізовані.

Суть трансформаційних методик полягає в перетворенні (трансформації) прототипу. При трансформації відомого технічного рішення використовується набір евристичних методів: кількісних змін модифікацій форми об’єктів або їхніх елементів, використовуваних матеріалів; інтеграцію або диференціацію; застосування способів просторово - часових перетворень і профілактичних заходів, тощо.

Більшість евристичних прийомів включає наступну інформацію, необхідну для програмування й використання методу: по-перше, вказується простір змінних, які варто змінювати в прототипі для одержання поліпшеного рішення; по-друге, встановлюється крок зміни змінних.

До трансформаційних методів належать, в першу чергу, алгоритми рішення винахідницьких завдань (АРВЗ) Г.С.Альтшулера, який систематизував фонд евристичних

прийомів і розробив алгоритм використання фонду. Разом з набором евристик і алгоритмів АРВЗ пропонується банк фізичних ефектів і явищ, а також методика роботи з ними. З роботами Г.С.Альтшулера перегукуються роботи Г.Я.Буша, що пропонує системний підхід до винахідництва й моделі пошуку ТР.

Евристичні методи дозволяють підвищити продуктивність праці проектувальника, однак не завжди здатні повністю задовольнити виконувані розробки новими ТР. Пошуки перспективних методів структурного синтезу, орієнтовані на використання ЕОМ, призвели до появи узагальненого евристичного алгоритму пошуку нових ТР.

Теорія евристичних рішень ще не може претендувати на широке застосування в САПР, однак подальша формалізація теорій використання евристичних прийомів і методів, заснованих на активній участі проектувальника в рішенні завдання, безсумнівно, призведуть до появи нових комп'ютерних програм.

Формалізовані методи вимагають задання деяким чином безлічі альтернативних варіантів структури, з яких потім синтезується шукане ТР. До формалізованих методів структурного синтезу належать:

- метод морфологічного ящика;
- метод "матриць відкриття " А.Маля;
- функціонально - вартісний аналіз і ін.

але найбільш визначеними і тому найбільш перспективними є методи морфологічного ящика: І-АБО-дерева, Р.Коллера і їхні

модифікації. Всі ці методи дозволяють вибрати або згенерувати деяку множину варіантів ТР, що задовольняє ТЗ, із заздалегідь підготовленого набору типових конструкцій.

Існує широкий набір конструктивних елементів для синтезу виробів (бібліотеки), які використовуються, як стандартні бібліотеки, в різних класах САПР.

Представлення трансформаційних і формалізованих методик у вигляді комп'ютерних програм вимагає залучення теорій штучного інтелекту, баз знань і графічних БД, застосування методів математичного програмування.

Більш просто зі згаданих методів реалізується метод морфологічного аналізу, що полягає в реалізації роботи з альтернативними проектами конструкцій. ОП досліджується, виділяючи ряд характерних ознак $f_1...f_n$. Після цього, для ознак визначаються градації, тобто взаємно виключні знання, однієї й тієї ж ознаки. Отримані результати оформляються в таблиці.

Виконання комірок морфологічних таблиць у вигляді графічних образів підвищує наочність і інформативність, полегшує роботу проектувальника. Розвиток методу йде по шляху пошуку скорочення процедур вибору ТР.

3.1 Параметричний синтез

При параметричному синтезі потрібно встановити, які параметри характеризують властивості режиму роботи об'єкта і які з них мають вирішальне значення. Параметри об'єкта проектування розподіляють на вхідні й вихідні, внутрішні й

зовнішні. Серед параметрів об’єкта проектування окремо виділяють показники ефективності - продуктивність, надійність, вартість, маса, габаритні розміри, точність та ін. Ці показники є кількісною оцінкою ступені відповідності об’єкта його цільовому призначенню.

Вхідні параметри встановлюються завданням на синтез об’єкта. Вихідні параметри визначаються в процесі проектування. Внутрішні параметри - це параметри елементів об’єкта. До них належать довжини й маси ланок механізму, величини опорів електричної схеми й т.д. Зовнішні параметри - це параметри зовнішнього стосовно об’єкта проектування середовища.

Для одержання заданих властивостей проектованого об’єкта потрібно задовольнити велику кількість, і найчастіше, суперечливих умов. Ці умови накладаються різними факторами, пов’язаним з експлуатацією об’єкта, технологією виготовлення і т.д. Із всіх висунутих умов звичайно вдається вибрати одну основну, інші приймаються в якості додаткові. Припустимо, основна умова проектування електричного фільтра - необхідна смуга проходження сигналу, додаткові - споживана потужність, розміри друкованої плати, вартість.

Основні умови параметричного синтезу можна записати у вигляді функції, що визначає вихідні параметри синтезу. Цю функцію називають цільовою.

Додаткові умови синтезу виражають у математичній формі звичайно нерівностями, що встановлюють припустимі

області існування параметрів синтезу, які задовольняють обмеженням.

Якщо не вдається відразу виділити одну основну умову синтезу, складають кілька цільових функцій, а потім йде спроба пошуку компромісного рішення.

3.2 Математичні моделі й методи параметричного синтезу

Вцілому задача параметричного синтезу зводиться до пошуку вихідних параметрів, при яких цільова функція приймає екстремальне значення [21]. Але на практиці практично завжди маємо наступний результат: специфіка ОП, відображена його ММ, і тип обраної цільової функції призводить до різних пошукових алгоритмів.

У статичних моделях залежність вектора вихідних параметрів від вектора вхідних і зовнішніх параметрів можна представити в явній формі або у вигляді системи алгебраїчних і трансцендентних рівнянь. У цьому випадку складний математичний апарат для рішення завдання не потрібен.

Якщо вихідні змінні задані як функції часу, а в рівняння входять похідні або інтеграли за часом, модель ОП описується системою диференційних або інтегральних рівнянь і називається динамічною.

Якщо змінні в моделі приймають значення з безперервної множини, ММ називається безперервною; якщо з конечної множини - дискретна (або логічна) ММ.

В аналітичних ММ шукані параметри явно виражені через відомі величини, а в алгоритмічних - ОП описується n - рівняннями з m - невідомих. Якщо

$$n=m,$$

задачу називають алгебраїчною і якщо вона описана лінійними рівняннями, то будемо мати одне рішення.

При $n>m$ задача є перевизначеною й практично взагалі не має рішення. Якщо $n<m$, задача недовизначена й має нескінченну безліч рішень. У проектуванні задача синтезу найчастіше зводяться саме до таких рішень. Тому найчастіше вони і вимагають методів оптимізації.

3.3 Методи оптимізації в проектуванні технічних систем

Будь-яка задача оптимізації має два характерних етапи [21]:

- постановка задачі;
- вибір методу рішення й рішення задачі.

На етапі постановки задачі формалізується поняття оптимізації, тобто формулюється цільова функція й обмеження на область припустимих рішень. Другий етап вирішить вибір методу рішення та проведе безпосереднє рішення задачі.

Перший етап виконується згідно формулювання та вимог ТЗ, відповідно до якого вибирається коректний, з погляду виконання ТЗ, критерій оптимізації.

Серед різнорідних величин вихідних параметрів не завжди вдається віддати перевагу одному з них, тому потрібна

побудова комплексного критерію, коли цільова функція поєднує декілька вихідних параметрів. Такі критерії називають глобальними. Якщо цільова функція приймає один з вихідних параметрів, то критерії називаються частками.

Вихідні параметри в цільовій функції можуть об'єднуватися різним чином, при цьому розрізняють наступні основні критерії: мультиплікативні, адитивні, мінімаксні (максимальні). Якщо при оптимізації не враховується статичний розкид параметрів, то відповідний критерій називається детермінованим, у противному випадку - статичним.

Виконання другого етапу оптимізації вимагає знання особливостей і можливостей численних методів, серед яких відсутній один універсальний.

Методи оптимізації класифікують за різними ознаками [10].

Залежно від типу критерію розрізняють методи безумовної й умовної, локальної й глобальної оптимізації.

Методи з деякими керованими параметрами називають методами багатомірного пошуку, при одному керованому параметрі - одномірного пошуку.

За способом пошуку рішення розрізняють методи випадкового, спрямованого й комбінованого пошуку.

При рішенні задач оптимізації можуть зустрічатися об'єкти з дискретними внутрішніми параметрами. Такі задачі вирішуються методами дискретного математичного програмування.

Методи оптимізації реалізовані в різних системах програмування й на різних обчислювальних машинах і поставляються фондами алгоритмів і програм у вигляді бібліотек прикладних програм.

4. Вибір раціональних варіантів рішення технічного завдання

Для пошуку рішень за допомогою евристичних прийомів у САПР широко поширені комбінаторні алгоритми, засновані на організації безлічі генеруємих варіантів у вигляді деревоподібних (ярусних) графів. Якщо, наприклад, кількість ярусів (параметрів) від 15 до 20, а число значень параметрів на кожному ярусі від 3 до 5, то кількість варіантів рішень по одному ТЗ становить від 10^8 до 10^12. Це означає, що навіть застосування ПК буде вимагати значних витрат машинного часу, якщо пошук ТР буде здійснюватися методом перебору.

Генерація множини рішень технічної задачі відноситься до області дискретного програмування. Для скорочення кількості проектованих варіантів застосовують наступні методи: відсікання (послідовний аналіз), гілок і меж, Парето, випадкового пошуку тощо.

Метод випадкового пошуку не гарантує відмінну від нуля ймовірність втрати оптимальних рішень.

Методи нелінійного й лінійного програмування вимагають настільки великої попередньої роботи з вивчення властивостей простору рішень, що найчастіше їхнє застосування є нераціональним.

Розглянемо найбільш раціональні методи вибору ТР рішення задачі, які найбільш використовуються при побудові саме різновидів САПР.

4.1 Послідовний аналіз

Найбільше розповсюдження через свою простоту одержав послідовний аналіз [21]. Основа - правило домінування. Сутність - відсікання безперспективних варіантів у процесі їхнього синтезу.

Нехай є 2 варіанти: і-й; j-й.

При цьому визначається зважена сума показників і- того (більше j- того, значить переважніше j- й). У результаті безперспективним є побудова нових варіантів рішення на основі варіантів показників і- того варіанта і тому гілка графу відсікається незалежно від рівня рішення задачі.

Розглянемо схему рішення. Нехай на (k-1) кроці пошуку рішень визначена безліч перспективних варіантів

$$x_{k-1}=(x_1^{(k-1)} , x_2^{(k-1)} \dots x_i^{(k-1)}),$$

кожний з яких характеризується k-параметрами.

На черговому k-му кроці пошуку кожний з отриманих раніше перспективних варіантів $x_i^{(k-1)}$ використовується для побудови нових варіантів рішення, число яких складе n_k ,

де n- число вершин деревоподібного графа.

Побудова виконується додаванням до $x_i^{(k-1)}$ одного (із тих, що ще не ввійшли в цей варіант) номера вершини, при

цьому, ці нові вершини характеризуються (k+1) - ми параметрами.

В результаті одержуємо розширену безліч варіантів рішень:

$$x_k = \{ x_1^{(k)} , x_2^{(k)} \dots x_{(n-k)}^{(k)} \}$$

Згрупуємо варіанти з xk, що закінчуються тим самим номером вершини деревоподібного графа, з однаковою множиною вхідних у послідовність номерів.

Використовуючи правило домінування, виділимо в кожній такій групі по одному кращому варіанту. Обрані варіанти утворять безліч x_k перспективних варіантів після k - того кроку пошуку. Процедура буде повторюватися доти, поки не буде виділений єдиний оптимальний варіант рішення.

Розглянемо роботу методу послідовного аналізу на прикладі. Нехай ми вже маємо декілька варіантів проведення траси, яка має з’єднати на ДМП два отвори під ЕРЕ. В якості основного обмеження будемо розглядати знаходження найкоротшого маршруту такого друкованого провідника (траси).

Побудуємо деревоподібний граф (вибір найкоротшого замкнутого маршруту від першої вершини до кінцевої вершини), як це показано на рисунку 2.

Для параметрів гілок переходу вибираємо значення:

C12 = C21 = 9 C13 = C31 = 10 C14 = C41 = 4

C23 = C32 =6 C24 = C42 = 8 C34 = C43 = 7

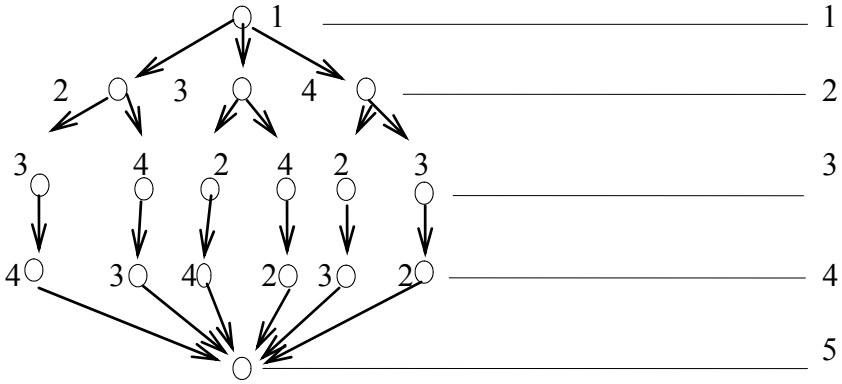


Рисунок 2 – Деревоподібний граф вибору найкоротшого замкнутого маршруту

Застосовуючи метод послідовного аналізу після другого кроку пошуку, коли три початкових варіанти визнаються перспективними, отримаємо рішення задачі. Дані говорять, що перспективним є варіант 1,4,3,2,1. Розрахунок параметрів приводиться в таблиці 1

Реалізація цього методу дозволяє скоротити кількість варіантів, що переглядають, в $(n-1)! : 2^n$ разів.

Недоліком методу послідовного аналізу є наступне. Для великої розмірності друкованої плати (більше 20 елементів) виникають труднощі з обчисленнями, пов’язані з значним їх обсягом. Окрім цього, при застосуванні правила домінування відсікається 50% безперспективних по даному рівню варіантів, серед яких може бути дійсно оптимальний варіант рішення задачі.

Таблиця 1 - Розрахунок параметрів одержаних варіантів рішення

x1	Оцінка	x2	Оцінка	x2	x3	Оцінка	x3	x4	Оцінка	x4	Оптим. варіант	$\hat{x}4$	Оптим. варіант
12 13 14 21 31 41 23 24 34 32 42 43	9 10 4 9 10 4 6 8 7 6 8 7	123 143 124 134 132 142	15 + 11 + 17 - 17 - 16 - 12 +	123 143 142	1234 1432 1423	22 - 17 + 18 +	1432 1423	14321 14231	26 + 28 -	14321	14321	14321	14321

4.2 Метод Парето

Для звуження області пошуку оптимальних рішень застосовується алгоритм виділення безлічі точок Парето. Визначення системи вагових коефіцієнтів W_1 і ранжирування по ній рішень із області Парето дозволяють одержати оптимальний компромісний варіант, збалансований за протиріччями щодо сукупності часткових критеріїв $Ф_1$ проектування або показників властивостей об’єкта.

Точками Парето є точки рішення XnE_x, для яких виконується умова:

$$\Phi(xn) = \sum_{i=1}^{m} Wi\ \Phi i\ (xn) <= \Phi(x)$$

Таким чином, точки області Парето являють собою перспективні варіанти рішення. Для існування області Порето, тобто для XnEx, необхідно , щоб існувала також безліч ваг:

$$W=(W_1,..., W_m)\ W_i >0$$

$$\sum_{i=1}^{m} W_i \frac{d\,\Phi}{d\,x_n} = 0$$

тобто grad (ΣWiΦi)=0, що дозволяє сформувати комплексну цільову функцію:

$$\Phi=\sum_{i=1}^{m} W_i\ \Phi_i$$

і мінімізувати її в програмі ~~для~~ одержання точок безлічі Парето.

Результати роботи методу виводяться у вигляді таблиць і визначають рішення, що відповідає розумному компромісу між всіма критеріями програмування.

Дана методика використовується для рішення питань багатокритеріальної оптимізації в основному при проектуванні ВЕТ (РЕА). Недолік: експертну оцінку кінцевим результатам дає група фахівців, інтелектуальний рівень яких впливає на кількість обраних варіантів та їх якість.

4.3 Метод гілок і меж

При вирішенні багатьох задач, пов’язаних із перебором існуючих варіантів рішень, ефективним є метод гілок і меж.

Скороченню перебору відповідає відсікання гілок дерева: чим ближче до кореневої вершини дерева відтинається гілка, тим ефективніше зменшується число варіантів, що переглядаються.

Метод полягає в наступному. Нехай ми маємо деревоподібний граф рішень (рисунок 2). Спочатку довільно вибирається який-небудь варіант, що відповідає на наш погляд найбільш оптимальному рішенню. У цьому випадку - це довжина шляху в графі рішень, що відповідає маршруту (1,3,2,4,1) з вагою

$$L=C13+C32+C24+C41=28 \text{ одиниць}$$

Обраний варіант є базовим і має вершини 1,3,2,4,1.

Далі розглянемо можливість використання інших початкових ділянок маршруту, що містять дві вершини, а саме (1,2) і (1,4). Для кожної ділянки послідовно обчислюється оптимістична оцінка нижньої границі маршруту або ваги відповідного варіанта. Якщо оцінка розглянутого варіанта більше базової, він вважається безперспективним і всі гілки, які його продовжують, відтинаються.

Далі переходимо до розгляду іншої ділянки. Якщо значення оцінки маршруту менше базової, то з кінцевої точки здійснюється розгалуження з наступним перебором ділянок, що містять тепер уже три вершини графа. Процес триває до одержання повного маршруту. Якщо його довжина виявиться менше базової, то отриманий оптимальний маршрут здобуває статус базового. Останній отриманий варіант і буде оптимальним.

Ефективність цього методу багато в чому залежить від способу одержання оптимістичних оцінок. При виборі початкової ділянки X1=(X1, X2, ...Xк) зводиться до наступного. З матриці відносин, яка характеризує граф рішень,викреслюються рядки

	1	2	…	N
1	C_{11}	C_{12}	…	C_{1N}
C_{ij}=2	C_{21}	C_{22}	…	C_{2N}
…	…	…	…	…
N	C_{1N}	C_{2N}	…	C_{NN}

з номерами (1,2,...,до-1) і стовпці з номерами (2,3,...к), обумовлені номерами гілок, що утворять розглянуту ділянку нового маршруту.

У частині, що залишилася, матриці Cij відшукується мінімальний елемент min Cij, який запам’ятовується. Потім, віднімаючи мінімальні елементи з інших ненульових елементів відповідних рядків, одержуємо нову матрицю Cij. На другому етапі - у кожному стовпці матриці відшукується мінімальний елемент, що також запам’ятовується й потім віднімається із всіх ненульових елементів відповідних стовпців.

При цьому оптимістична оцінка довжини маршруту визначається за формулою

$$\Delta = L(1,2,...k) + \sum_{i=1}^{N} \min C_{ij} + \sum_{j=1}^{N} \min C_{ij} \qquad (1)$$

Для графа рішень, зображеного на рисунку 2, матриця відношення має вигляд:

	1	2	3	4
1	_	9	10	4
2	9	_	6	8
3	10	6	_	7
4	4	8	7	_

Якщо в якості початкового обрана ділянка C_{12}, то відповідно до процедурної оптимістичної оцінки маршруту одержуємо:

	1	2	3	4	min C_{ij}
1	_	_	_	_	_
2	9	_	8	6	6
3	10	_	0	7	7
4	4	_	7	0	4

	1	2	3	4
1	_	_	_	_
2	3	_	0	2
3	3	_	0	0
4	0	_	0	0

у результаті чого:

$$\Delta C12=C12 + \Delta' \min C12 + \Delta'' \min Cij = 9+6+7+4=26$$

Аналогічно розраховуються інші ділянки маршруту.

Послідовність подальшої роботи алгоритму методу гілок та меж для графа на рисунку 2 наведена на рисунку 3.

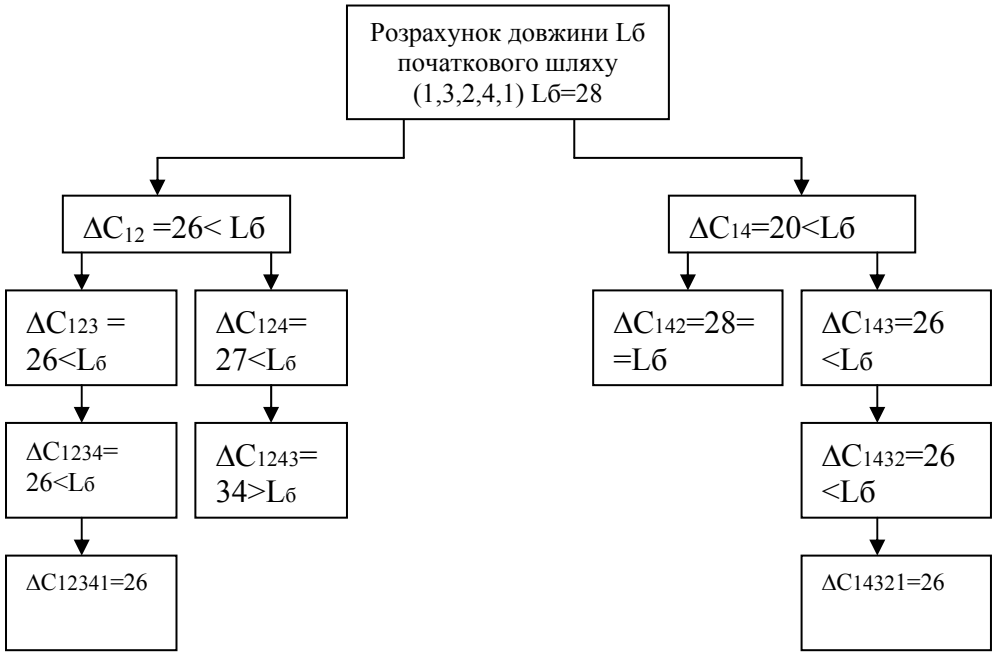


Рисунок 3 – Алгоритм роботи методу гілок та меж

Відзначимо, що два отриманих рішення є оптимальними. У цьому випадку, вибирається одне з них.

БАЗИ ДАНИХ В САПР

1 Загальні положення. Структура інформаційного забезпечення

2 Банк даних і база даних в САПР

3 Етапи розвитку баз даних

4 Рівні абстракції баз даних

1. Загальні положення. Структура інформаційного забезпечення

Як ми вже знаємо, дані або інформація - важливий ресурс будь-якої системи. За структурою - інформаційне забезпечення [11] [13] САПР - це складний комплекс засобів, що забезпечує своєчасну, повну й достовірну інформацію усіх етапів (рівнів) проектування.

Ядро ІЗ - банк даних, який утримує одну або кілька баз даних (БД) і СУБД. До основних задач ІЗ САПР віднесемо наступне:

- перетворення вхідної інформації в необхідну вихідну;

- формування масивів інформації до вигляду, зручного для зберігання, пошуку й видачі даних.

Функції ІЗ:

- експлуатація інформаційних файлів БД САПР;

- збирання і зберігання даних;

- робота в автоматичному режимі в якості НСІ;

- виведення інформації по вимозі користувача на екран монітора, друкуючий пристрій тощо чи передача інформації між прикладними програмами.

Для ІНСАПР характерне спільне використання єдиних даних різними ПРОПМ (паралельне обслуговування), що викликає необхідність формування загальної БД. Наприклад, для комплексу програм функціонально-логічного проектування топології друкованих плат та для забезпечення сумісності між цими програмами, програмами аналізу і топологічного проектування, важливе значення набув автоматичний взаємозв’язок між ними.

Приклад ІНСАПР із паралельною структурою обслуговування єдиною БД ПРОПМ наведений на рисунку 1.

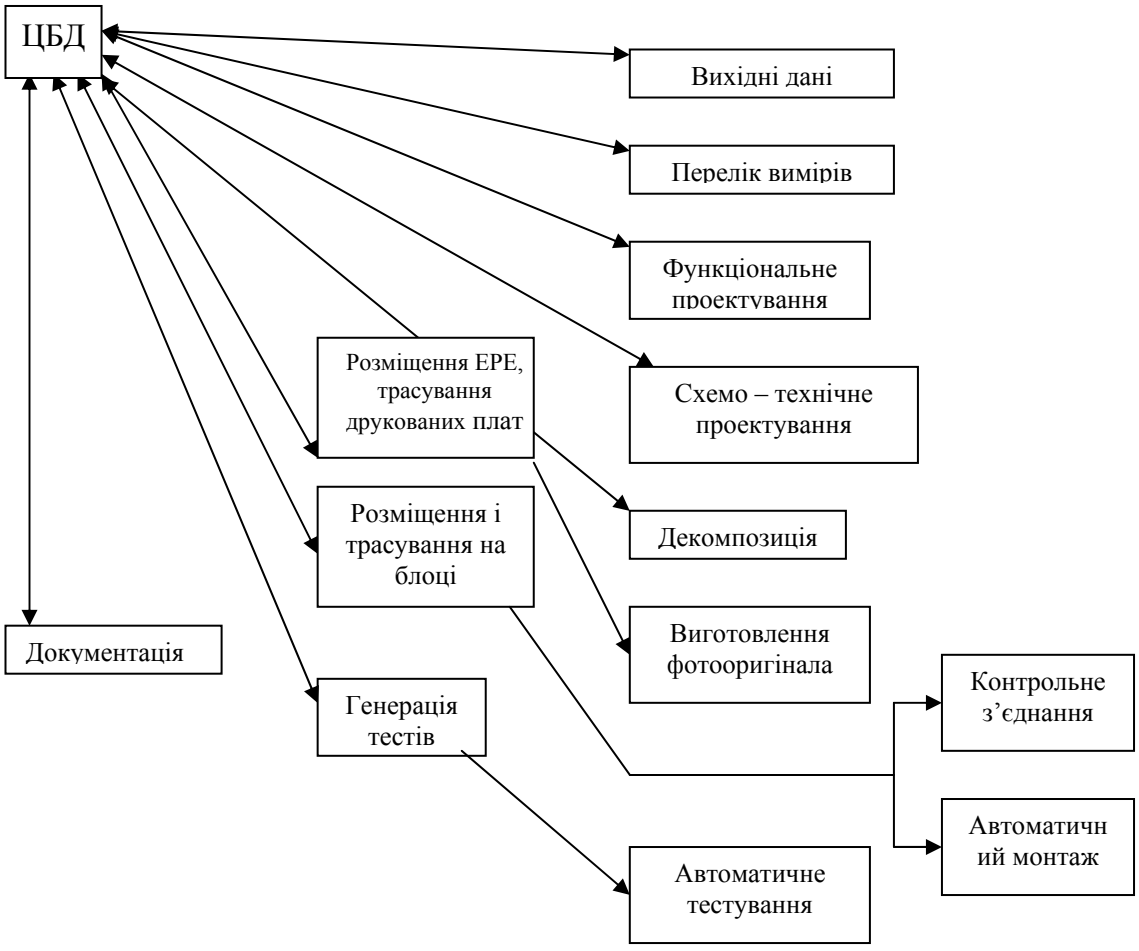


Рисунок 1 – Структурна схема ІНСАПР з паралельним обслуговуванням базою даних різновидів ПРОПМ

Загальна БД для паралельного використання декількома програмами стимулювала розвиток стандартів на вихідні дані всіх програмних модулів, тобто - було реалізоване уніфіковане подання даних (УПД). Крім цього, була проведена побудова ефективних структур даних, але виникла необхідність введення санкціонованого доступу до БД з цілю забезпечення захисту інформаційних ресурсів від несанкціонованого доступу та використання.

2 Банк даних і база даних в САПР

Так яка ж роль БД у САПР? Насамперед, вона забезпечує зв'язок в єдину систему окремих програмних модулів [23].

САПР із багаторівневим проектуванням додали в структуру БД третій вимір. Замість простого поділу проекту на доступні для огляду частини (ІНСАПР), ієрархічна структура диференціює ці частини по класах. Результати проектування на рівні, наприклад, генерації варіантів рішення, стають довідковими даними для більш високого рівня, наприклад, вибору варіанта рішення. Тобто, вводитися висхідна (спадна структура) БД, або так звана ієрархічна структура побудови БД.

Банк даних - це накопичена інформація, організована з урахуванням наступних вимог, обумовлених чисто специфікою САПР. А саме:

- інтегрування вихідних даних або централізація масивів (БД);

- забезпечення централізованого доступу до масивів;

- динамічне зберігання даних;

- відсутність дублювання даних;

- незалежність банку даних від спеціальних випадків застосування;

- мультиплексний і монорежим користування;

- представлення банку даних у вигляді БД і СУБД.

СУБД – це комплекс управляючих БД чи банком даних програм. Перераховані особливості банку даних обумовили основний принцип його побудови - структуризація інформації.

Структуризація - це розбиття масиву інформації на окремі сегменти даних, які логічно пов’язані між собою, наприклад - умістом коду. Цим було вирішене питання тісного зв’язку між масивами інформації, що дозволяють їх використовувати в будь-якому варіанті. Структурна організація даних у масивах - основний критерій, що характеризує банк даних.

Структурна модель єдиної інформаційної БД для якоїсь сукупності розв’язуваних задач наведена на рисунку 2.

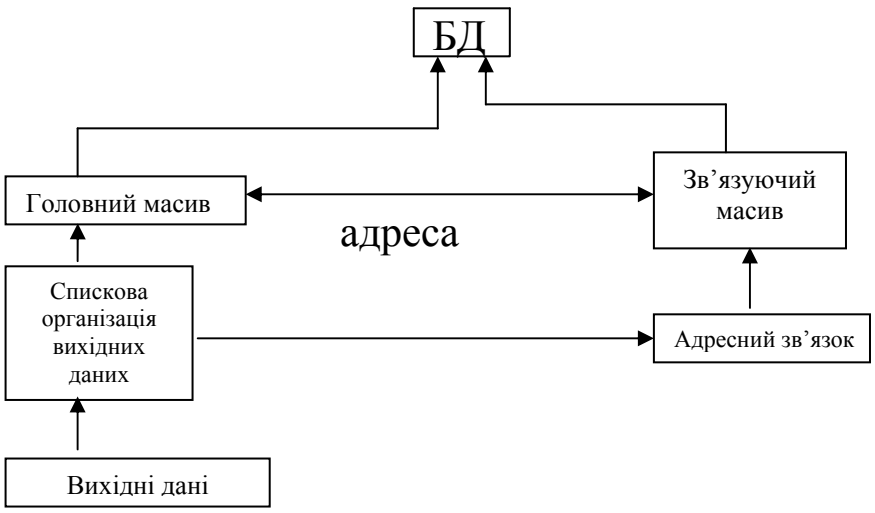


Рисунок 2 - Висхідна структура побудови БД

А тепер розглянемо що таке адресний зв’язок.

Приклад - елементарний діалоговий режим. Адресний зв’язок реалізовується за допомогою кодів. Нехай нам необхідно об’єднати 6 масивів інформації за допомогою коду, тобто організувати адресний зв’язок.

Масиви вихідних даних		**Код**
КНТУ - університет	Кількість символів	1
МТФ – факультет	коду, привласнених	1
ПЗІ - кафедра	кожному масиву	01
КІ	вихідних даних	02
Студент		001
Викладач - професія		002
Іванов І.І - П.І.Б.		001
Іванов І.П.		002

Структура коду - 10 символів. Таким чином, студент Іванов І.І., що навчається на кафедрі ПЗІ МТФ КНТУ буде мати наступний код в БД: 1101001001

Тобто, утворився ланцюжок для швидкого знаходження потрібних даних у величезному масиві інформації.

3. Етапи розвитку баз даних в САПР

У розвитку САПР за останні 10 років можна виділити два етапи, пов’язані із проблемами побудови БД.

Перший етап – побудова САПР із єдиною БД [21], [23], як це показано на рисунку 3.

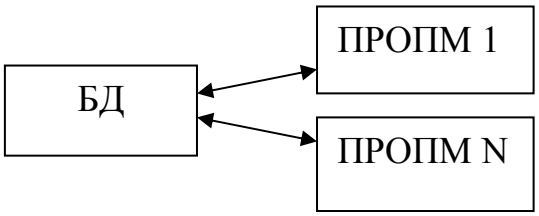


Рисунок 3 – Структурна схема САПР з єдиною БД

Таку організацію БД мають майже всі розроблені на сьогодення ІНСАПР. Недолік - відсутність інформації про конкретний уміст БД, тому виникла необхідність проводити інтегрування незалежних БД і ПРОПМ, як це показано на рисунку 4.

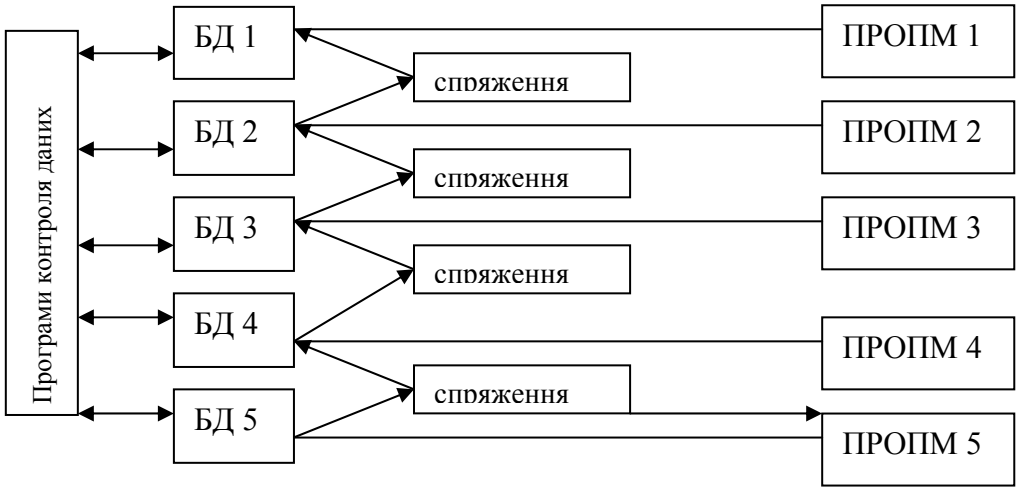


Рисунок 4 – Структурна схема інтегрування незалежних БД та ПРОПМ

Проте в таких системах, із-за використання мультиплексного режиму роботи користувачів, цілісність даних не гарантується. Це є серйозним недоліком і вимагає розробки та введення до складу вищеозначених систем програм контролю даних.

Створення такої БД дуже трудомісткий процес, найчастіше контроль і переробка даних забирає більше обчислювальних ресурсів, ніж виконання власне проектування. Вихід можна знайти в частковому об'єднанні БД, як це показано на рисунку 5.

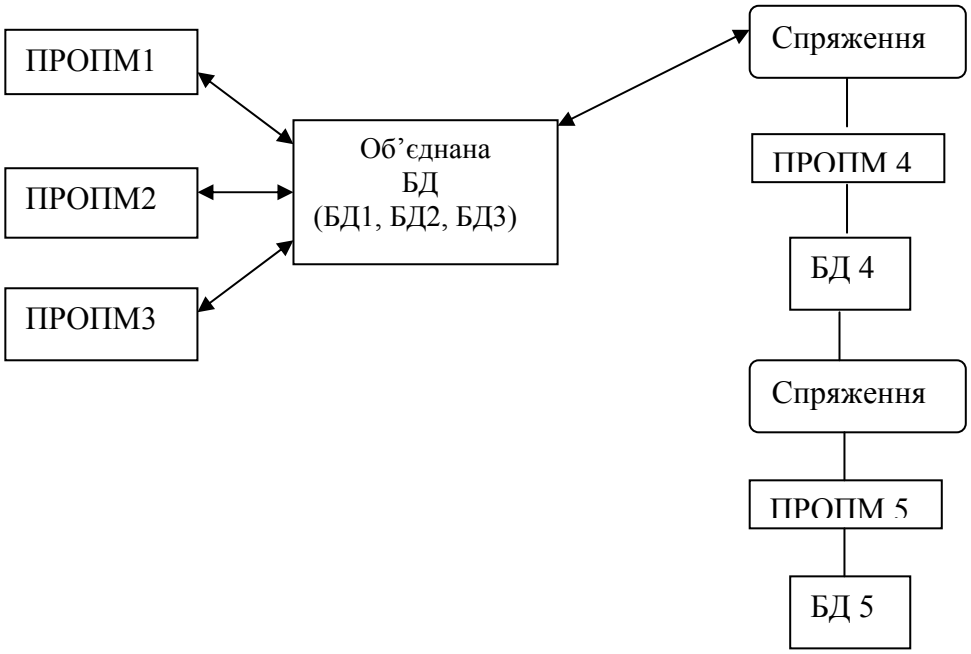


Рисунок 5 – Структурна схема САПР з частковим об'єднанням БД

Однак загроза втрати інформації (в основному через людський фактор) повністю не ліквідується й при наявності часткового об'єднання БД. Була потрібна зовсім нова структура БД.

Другий етап – пов'язаний з побудовою централізованої БД (ЦБД) спільно з допоміжними БД.

Головна роль СУБД полягає в забезпеченні користувача інструментами, що дозволяють оперувати даними в абстрактних термінах (ЕРЕ, друкованих плат і т.д.), не зв'язаних зі способами їхнього зберігання в ЕОМ. Крім того,

СУБД повинна виконувати наступні функції: забезпечення санкціонованого доступу користувачів в залежності від їх пріоритету; дані повинні відповідати деяким властивостям компонентів ОП (наприклад, у ЕРЕ повинна бути обмежена кількістю провідників, підведених до їх виводів і т.д.); синхронізацію та забезпечення режиму мультиплексного доступу користувачів; захист від відмов і відновлення.

У своїй головній ролі СУБД подібна до транслятора (інтерпретатора) з мови дуже високого рівня (мови опису ОП), засобами якого можна ідеально специфікувати проект.

4. Рівні абстракції БД в САПР

Між рівнем представлення даних користувачем і рівнем представлення даних, зрозумілих для ЕОМ (рівень бітів, де біти - одиниця інформації виражена у вигляді двійкового представлення одного розряду), існує кілька рівнів абстракції даних [19]. Існування таких рівнів спрощує перехід від одного верхнього рівня до самого нижнього. У реальних СУБД існує три рівні абстракції баз даних, як це показано на рисунку 6.

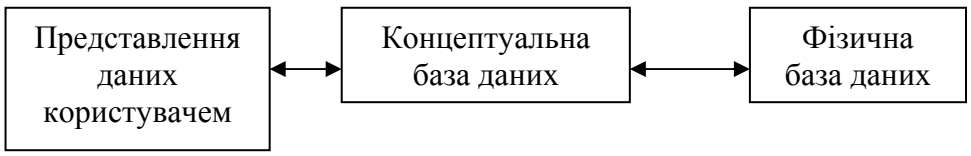


Рисунок 6 – Рівні абстракції СУБД

Відзначимо, що реально зберігається тільки фізична база даних, концептуальна БД - це абстрактне відображення БД, а представлення даних користувачем - це абстракції деяких частин концептуальної БД.

Розходження в рівнях абстракції між представленнями й концептуальною БД невелике. І та й інша БД мають справу з абстракціями такого роду, як «резистор, транслятор тощо» і з такими абстрактними зв’язками, як «провести аналіз операційного підсилювача тощо». Фізична БД перебуває на самому нижньому з розглянутих рівнів абстракції. Найчастіше вона є складовою довготермінових САПР.

Концептуальна БД містить мови визначення даних, що дозволяють записувати концептуальне представлення даних у термінах деякої «моделі даних».

CAD, CAM, CAE-ТЕХНОЛОГІЇ АВТОМАТИЗОВАНОГО ПРОЕКТУВАННЯ. ЗАВДАННЯ МОНТАЖНО-КОМУТАЦІЙНОГО ПРОЕКТУВАННЯ

1 Загальні положення

Як правило, вироби РЕА (ЗОТ) являють собою комплекс конструктивних модулів, об’єднаних механічними, електричними й іншими зв’язками, які виконують задані функції відповідно до цільового призначення. Тобто - РЕА (ЗОТ) – це багаторівнева ієрархічна структура, що складається з п’яти рівнів:

- рівень 0 утворюють неподільні складові, такі як ЕРЕ та мікросхеми;

- рівень 1 включає мікрозборки, мікромодулі й інші об’ємні базові моделі;

- рівень 3 включає блоки та конструктивно закінчені складальні одиниці;

- рівень 4 утворять функціонально й конструктивно закінчені вироби електронної техніки.

У якості РЕА (ВЕТ, ЗОТ) може виступати складова будь-якого рівня. Це й операційний підсилювач, виконаний у вигляді інтегральної схеми, і однокристальна ЕОМ, і магнітофон для запису, і система спостереження за супутником.

Завдання конструювання виробів РЕА (ЗОТ) прийнято розглядати як монтажно-комутаційне проектування всіх рівнів, підрозділене на два типи: просторове конструювання й плоскі конструкції [21].

2 Просторове конструювання

Якщо компоненти й з'єднання конструкції розташовуються в N-му числі площин, то таку конструкцію прийнято називати просторовою. До складу просторової конструкції входить N-а кількість плоских конструкцій і найчастіше, якщо просторова конструкція не має високої ступені складності, її розглядають при проектуванні, як деяке сумарне число плоских конструкцій.

На сьогодення в інженерній практиці широко використовується ряд програм просторового конструювання, в основі яких лежить реалізація різних модифікацій методу кінцевого елемента.

Задачі просторового конструювання прийнято описувати рівняннями в частинних похідних, які ефективно вирішуються за допомогою методу кінцевого елемента, застосування якого надзвичайно широке: машинобудування, будівельна справа, гідродинаміка, аеродинаміка, нова техніка, гірнича справа, моделювання, математична фізика тощо [21], [29]. Цей метод

ефективний при рішенні задач прогнозування статичного, динамічного й температурного поводження машин і механізмів; при оцінці напруг і деформації конструкцій; розрахунків на міцність конструкцій.

2.1 Метод кінцевого елемента

Сутність методу кінцевого елемента зводиться до наступного [21]. Область визначення рівнянь, яку знаходять, у частинних похідних покривається деяким кінцевим числом непересічних один з одним елементів і шукається дискретна апроксимація рішення рівнянь у вигляді лінійної комбінації

$$\sum \alpha_i \varphi_i$$

деяких базисних функцій $\varphi_i(x)$. Останні пов'язані з окремими введеними елементами й можуть бути частково поліноміальними, раціональними й ін.

Параметри цих базисних функцій підбираються таким чином, щоб забезпечити необхідну ступінь безперервності функцій, що описують сусідні елементи. Апроксимація рішення по всій області визначення рівнянь може бути виражена через значення базисних функцій і їхніх похідних у вузлових точках, при цьому кожна функція φ_i дорівнює нулю на більшій частині області визначення й відмінна від нуля тільки в околиці одного вузла. У загальному вигляді апроксимуюче рішення функція має вигляд

$$\upsilon(\chi) = \sum_{i=1}^{N} \left[\alpha_i(\chi)\phi_i + \beta_i(\chi)\left(\partial\upsilon / \partial\chi_1 \right)_i + \cdots \right], \quad (1)$$

де N-число вузлів сітки елементів, що покривають область визначення функції.

У більшості практичних модифікацій метод кінцевого елемента (1) спрощується до вигляду

$$\upsilon(\chi) = \sum_{i=1}^{N} \alpha_i(\chi)\phi_i \qquad (2)$$

Знаходження коефіцієнтів $\alpha_i(\chi)$, а в загальному випадку $\beta_{i(\chi)}\gamma_{i(\chi)}$ є найбільш критичною й самою складною процедурою методу кінцевого елемента. Зазвичай вони визначаються за допомогою додаткової оптимізаційної процедури, пов'язаної з поданням вихідної задачі в деякому еквівалентному варіаційному формулюванні.

Наприклад, у методі Ритца для знаходження коефіцієнтів формулюється система лінійних рівнянь наступного вигляду

$$\frac{\partial}{\partial\alpha_i} I\left(\sum_{i=1}^{N}\alpha_i\phi_i\right) = 0,\ \mathrm{i} = 1 \qquad (3)$$

де, відповідно до принципу варіації, використовується подвійний інтеграл по границі ∂R області визначення R:

$$I(u) = SR\ S\ F\ (x, y, u, u_x, u_y)\ d_x d_y, \qquad (4)$$

яка має екстремум тільки в тому випадку, коли шукана функція двох змінних u(x,y) задовольняє диференціальне рівняння Ейлера-Лангранжа типу

$$\partial/\partial\chi\, F_{ux} + \partial/\partial y\, F_{uy} - F = 0 \qquad (5)$$

Істотним при варіаційній постановці задачі є спосіб задання крайових умов. Розрізняють умови Даричле або геометричні однорідні й неоднорідні умови

$$U(0) = 0,\ U = g \tag{6}$$

і умови Неймана (природні динамічні крайові умови)

$$\partial u / \partial n = 0, \tag{7}$$

де n - зовнішня нормаль до границі області визначення функції ∂R.

В інженерній практиці найбільш відомі наступні САПР, які реалізують просторові конструкції:

- SAFE - програма моделювання біполярних МОП-транзисторів. Розроблювач - фірма IBM;

- ПРОЧНОСТЬ - для проведення розрахунків на міцність авіаційних і інших конструкцій;

- ЛАДА - проектування залізобетонних конструкцій підземних і наземних споруджень у промисловості й цивільному будівництві (Київ) і інші.

3 Плоскі конструкції

Найпоширеніші в монтажно-комутаційному проектуванні плоскі конструкції, у яких з'єднання й компоненти розташовуються в обмеженому числі паралельних площин.

Прикладом плоских конструкцій є друковані монтажні плати [21], [29]. Щоб таку плату реалізувати, необхідно задовольнити ряд обмежень, які розподіляють на метричні й топологічні.

Метричні обмеження визначаються:

- фіксованими розмірами компонентів;

- шириною друкованих провідників;

- мінімально припустимими відстанями між різними елементами конструкції й друкованими провідниками ;

- габаритними розмірами конструкцій.

До топологічних обмежень відносять:

- заборонене розташування трас у заданих областях монтажного простору;

- заборонене перетинання різних з’єднань;

- вимога електричного об’єднання певних контактів.

4 Монтажно-комутаційне проектування

Процес проектування можна розподілити на два послідовно виконуваних етапи. На першому етапі виконується побудова попереднього варіанта розміщення компонентів і з’єднань, яке задовольняє вимоги топологічних обмежень.

На другому етапі отриманий варіант розміщення перетвориться в реалізовану монтажну схему, яка задовольнить не тільки метричні, але й топологічні обмеження і вимоги ТЗ. Цей підхід до проектування плоских конструкцій називається топологічним і включає наступні основні етапи:

- побудова математичної моделі (ММ) ОП;

- формування плоского укладання моделі об’єкта (топологічний аналіз);

- побудова попереднього варіанта розміщення компонентів і з’єднань;

- побудова остаточного варіанта конструкції.

На першому етапі будується ММ, що відображає розглянуті властивості ОП, на другому - виділяється

максимальна частина моделі, що укладається на площині без порушення топологічних обмежень. Якщо виділена частина містить не всі елементи моделі, то виконується третій етап, на якому вихідна (об'ємна) модель перетвориться в планарну (плоску). Четвертий етап не обов'язковий, хоча й забезпечує розроблювача попереднім розміщенням елементів. Остаточний варіант розміщення компонентів і з'єднань реалізується на заключному етапі.

Топологічне проектування застосовується не тільки для реалізації монтажно-комутаційних виробів РЕА (ЗОТ), але й для проектування інженерних мереж будівельних об'єктів, компонування обладнання цехів і прокладення монтажних з'єднань, проектування транспортних трас і т.д.

При рішенні конкретних завдань використовуються різні ММ і їхні конструкції у вигляді графів, що імітують схеми з'єднань.

4.1 Поняття теорії графів

Дамо визначення основним поняттям теорії графів.

Граф $G = (V, E, F)$ – це сукупність трьох об'єктів:

- безліч вершин V;

- безліч ребер E;

- інцидентор F вказує, які пари вершин якими ребрами з'єднані.

Два ребра із загальною вершиною називаються суміжними. Якщо порядок вершин у парі (V_1, V_2) чітко не

означений, то ребро називають ланкою, у противному випадку - дугою.

Кілька однотипних ребер (ланок або дуг) між однією парою вершин вважають паралельними.

Неограф - граф, всі ребра якого вважають ланками (неорієнтований граф).

Орграф (орієнтований) - граф, всі ребра якого дуги.

Змішаний граф - серед ребер - і дуги, і ланки.

Орієнтація ребра - заміна ребра дугою.

Дезорієнтація ребра - заміна дуги ланкою.

Уніграф - граф, у якого кожній парі вершин відповідає не більше одного ребра.

Мультиграф - якщо є хоч одна пара вершин, який відповідає кілька ребер.

Підграф - частина графа, що включає всі ребра з E (G), обидві вершини яких належать V.

Суграф – частина графа з підмножиною вершин $V_1 = V_i$ безліччю ребер F_1.

Повний граф - містить всі можливі ребра.

Зірковий граф – безліч V складається з деякої вершини й суміжних з нею вершин.

Цикл - замкнутий шлях графа.

Деревовидний граф циклів не утримує.

Точка зчленування - вершина, видалення якої збільшує число його компонентів зв’язності.

Блок графа - максимально зв’язний підграф без точок зчленувань.

Нероздільний граф - видалення будь-якого ребра не призводить до збільшення його компонентів зв'язаності.

Планаризація графа - подання сукупністю планарних суграфів з непересічною безліччю ребер.

4.2 Графові моделі

Найбільш часто використовуються наступні графові моделі: граф, гіперграф і ультраграф. В цих моделях вершинам графів відповідають компоненти ОП.

В моделях у вигляді графа кожне з'єднання відображене сукупністю ребер, з'єднуючих відповідні вершини. Граф будується відповідно до функціональної схеми ОП. Приклад функціональної схеми, побудованої на основі схеми електричної принципової, показаний на рисунку 1, а її моделі у вигляді простого графа - на рисунку 2.

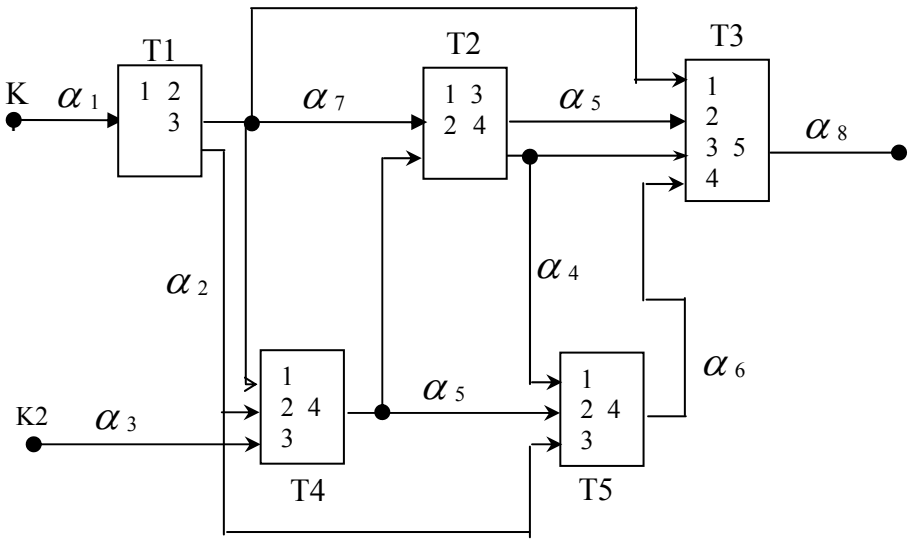


Рисунок 1 – Функціональна схема об'єкту проектування

У різних типах графових моделей використовуються різні сукупності таких ребер. Наприклад, у гіперграфі кожне

з’єднання (ланка) представляється гіперребром. Гіперребро - поняття узагальнене, являє собою підмножину із двох елементів, що належать графу: безліч вершин *V*; безліч ребер *E*.

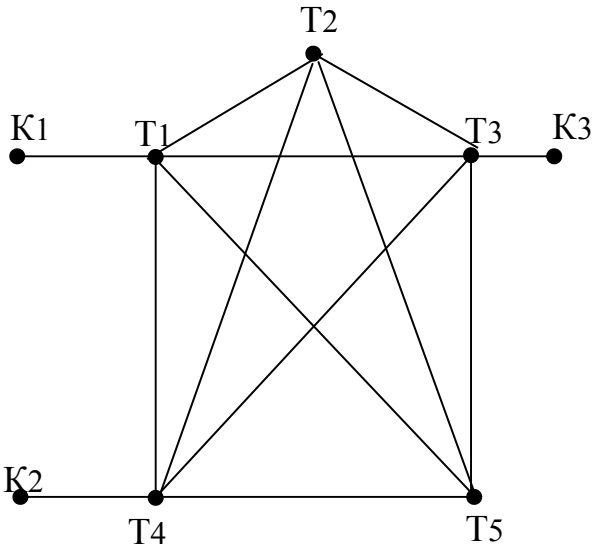


Рисунок 2 – Графова модель об’єкту проектування

Графічно ребро зображується однозв’язною областю, усередині якої знаходяться відповідні точки першої множини (заміна паралельних ребер однозв’язною площиною).

У багатьох випадках використовують дводольний біхроматичний граф Кенінга, що відповідає гіперграфові і являє собою граф із двома непересічними безлічами вершин відповідного компонента і гіперребер.

Графічне зображення гіперграфа показана на рисунку 3.

В ультраграфах кожне ребро має орієнтацію, що вказує на джерело й споживача сигналів (рисунок 4).

У більш точних моделях ОП вершини графових моделей відповідають окремим частинам компонентів (наприклад, їхнім контактам). Такі моделі використовуються для рішення задач трасування з’єднань між контактами ЕРЕ.

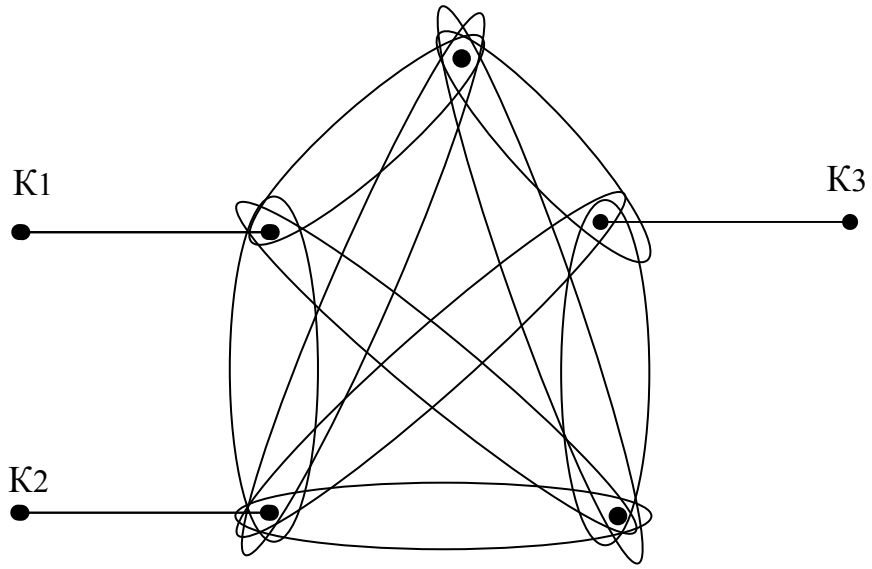


Рисунок 3 – Гіперграф ОП

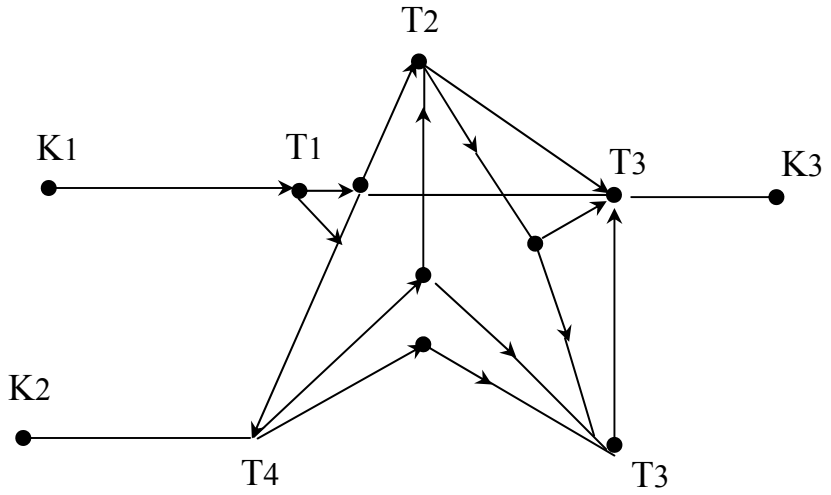


Рисунок 4 – Ультраграф ОП

Опис ОП завжди має ієрархічну структуру [21], моделлю якої є орієнтоване кореневе дерево. Це пов’язане із процесом послідовної побудови більше детальних моделей об’єкта проектування. Таким чином, конструкторські задачі

допускають дворівневий або трирівневий розгляд. При дворівневому поданні, наприклад, компонент (i + 1)-го рівня розглядається, як весь ОП, а компоненти і-го рівня, як компоненти цього об’єкта. Подання ОП у вигляді сукупності блоків різного рівня визначає формальну структуру його моделі. Відповідна сукупність геометричних моделей (графів) визначає структурну модель його конструкції. Як функціональний, так і конструктивний розподіл ОП має ієрархію.

Таким чином, ієрархічна структурна модель ОП і його конструкції, реалізовані розподілом усього об’єкта на блоки різного рангу, забезпечує зручність проектування, виготовлення й експлуатації і є необхідною умовою для застосування методів автоматизації.

Постановка й рішення конструкторських задач неможливі без визначення ММ монтажного простору для кожного рангу блоків.

5 Монтажний простір

Монтажним простором блоку і-го рангу називається область, обмежена габаритами цього блоку. Монтажний простір є метричним простором, у якому розміщаються блоки (і-1)-го рангу й здійснюється їхня комутація.

В інженерній практиці більшість конструкторських задач пов’язана з конструюванням у тривимірному монтажному просторі. Разом з тим специфіка цих задач і особливостей конструювання звичайно дозволяють перейти до адекватного

задання у двомірному монтажному просторі за рахунок розгляду проекцій компонентів на площину.

Розрізняють безперервні й дискретні монтажні простори.

Дискретний монтажний простір характеризується кінцевим числом заздалегідь заданих позицій для розміщення компонентів. Дискретний монтажний простір може бути регулярним і нерегулярним. У регулярних - позиції розташовуються з постійним кроком по вертикалі й горизонталі. Нерегулярні мають блоки середніх і молодших рангів (наприклад, у цифровій апаратурі).

У безперервних монтажних просторах не можна заздалегідь визначити координати позицій, адже при розташовувані компоненти мають різні розміри й форму.

При здійсненні комутації компонентів також будуються спеціальні монтажні простори, як дискретного регулярного типу (наприклад, у вигляді дрібної ортогональної технологічної сітки), так і нерегулярного (у вигляді графа каналів у випадку провідного монтажу в багатошаровій друкованій платі). У більш складних алгоритмах використовують безперервний монтажний простір, а в більш простих – дискретний монтажний простір, який зазвичай називають координатною сіткою.

У випадку багатошарового монтажу монтажний простір утвориться шляхом спеціального об’єднання монтажних просторів окремих шарів ОП.

Оскільки основні завдання конструювання відносяться до поліноміально повних проблем, для їхнього рішення використаються евристичні (наближені) алгоритми.

Основні задачі конструювання, названі на початку лекції, а алгоритми їхнього рішення ми розглянемо в наступних лекціях.

КОМПЛЕКС ЗАСОБІВ АВТОМАТИЗОВАНОГО ПРОЕКТУВАННЯ: ЛІНГВІСТИЧНЕ, ПРОГРАМНЕ, ІНФОРМАЦІЙНЕ, ТЕХНІЧНЕ, МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ САПР

Вступ

1 Лінгвістичне забезпечення САПР

1. 1 Класифікація мов САПР

1.2 Діалогові мови

1.3 Організація діалогу в САПР

1.4 Діалогові обміни

1.5 Способи взаємодії людини і комп’ютера

2 Програмне забезпечення САПР

2.1 Склад ПЗ

2.2 Класифікація ПЗ САПР по функціональному призначенню

2.3 Основні принципи проектування ПЗ САПР

2.4 Модульний принцип побудови програм

3 Інформаційне забезпечення САПР

3.1 Банки даних (БнД)

3.2 Інформаційні потоки в САПР

3.3 Функціональний розподіл БД

3.4 Структура БД (моделі даних)

4 Технічне забезпечення САПР

Вступ

Структурні елементи САПР можна згрупувати по видах забезпечення автоматизованого проектування. Таких забезпечень сім: технічне, програмне, інформаційне, математичне, методичне та організаційне.

Перші п’ять видів забезпечень далі розглянемо більш детально, коротко зупинившись на двох останніх - методичному та організаційному забезпеченнях.

Методичне забезпечення САПР складають документи, що входять до її складу, регламентують порядок її експлуатації, характеризують склад і правила обирання засобів автоматизованого проектування. При цьому, технічна документація, яка регламентує процес розробки САПР, не входять до складу методичного забезпечення. Документи методичного забезпечення носять в основному інструктивний характер.

Організаційне забезпечення САПР включає наступні документи: положення, інструкції, накази, штатні розклади, кваліфікаційні вимоги до обслуговуючого персоналу та користувачів і інші документи, що регламентують організаційну структуру підрозділів проектної організації і взаємодію підрозділів з комплексом засобів автоматизованого проектування.

Взаємозв’язана сукупність програмного, інформаційного і методичного забезпечення, призначена для отримання закінченого проектного рішення або виконання уніфікованих процедур, називається програмно-методичним комплексом (ПМК).

Останнім часом програмно-методичні комплекси САПР включають, разом з програмами і базами даних, в комплекти документації, які є частиною методичного забезпечення САПР.

Однак, допускається і більш ширше використання поняття методичного забезпечення, при якому під методичним забезпеченням розуміється сукупність математичного, лінгвістичного забезпечень і названих вище документів, що реалізовують правила використання засобів проектування під час промислової експлуатації за безпосереднім призначенням..

За призначенням підсистеми розділяють на ті, що виконують проектні процедури і обслуговуючі. Підсистеми проектування мають об’єктну орієнтацію і реалізують певний етап (стадію) проектування або групу безпосередньо пов’язаних проектних задач. Підсистеми обслуговування мають загальносистемне застосування і забезпечують підтримку функціонування підсистем проектування, а також оформлення, передачу і виведення отриманих в результатів за вимогою користувача.

1. Лінгвістичне забезпечення САПР

Лінгвістичне забезпечення САПР - це сукупність мов, які використовуються в САПР для подання інформації про

проектовані об’єкти, процес і засоби проектування, якою обмінюються люди з ЕОМ і між собою в процесі автоматизованого проектування.

Лінгвістичне забезпечення покликане спростити і полегшити як процес розробки систем автоматизованого проектування, так і взаємодію з нею користувача.

1.1. Класифікація мов САПР

Мови САПР діляться на два класи: мови програмування, мови проектування, як це показано на рисунку 1.

Мови програмування призначені для складання і запису програм і, разом з операційною системою ЕОМ, служать основним засобом для розробки програмного забезпечення САПР. Мови програмування включають: машинні коди, асемблери, мови високого рівня.

Машинний код - це мова, пропозиції якої подібні до машинних команд, записаних не в двійковій, а у вісімковій або шістнадцятирічній системі зчислення. Мова асемблера - автокод, розширений макрокомандами, виразами, засобами, що забезпечують модульність програм. Текст програми з мови асемблера компілятором перекладається в машинні команди.

Використання машинно-орієнтованих мов дозволяє досягати найвищої ефективності об’єктних програм з точки зору витрат обчислювальних ресурсів, машинного часу і пам’яті. Ці мови універсальні в сенсі застосування до рішення задач різних класів - науково-технічних і економічних, системних і прикладних. Проте програмування на цих мовах

вимагає високої кваліфікації програміста і призводить до збільшення термінів розробки прикладного програмного забезпечення.

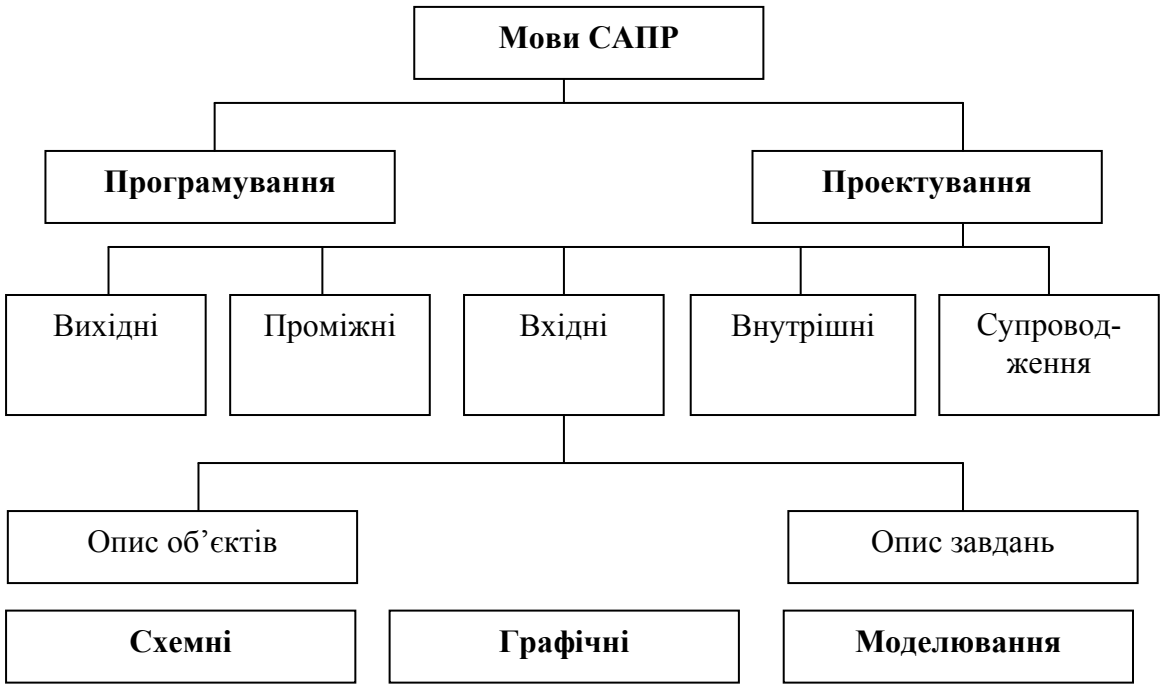


Рисунок 1 – Класифікація мов лінгвістичного забезпечення

Головні недоліки машинно-орієнтованих мов полягають в наступному: низька продуктивність створення програм і складність перенесення програм на ЕОМ з системою команд, відмінною від тієї, на яку орієнтована мова.

Алгоритмічні мови високого рівня - основний засіб розробки прикладного програмного забезпечення. У САПР найбільше поширення отримали мови DELPHI, C++, C тощо.

Мови програмування - мови, призначені для опису ОП. Вимоги до мов програмування: зручність використання; універсальність, ефективність об’єктних програм.

Зручність використання виражається у витратах часу програміста на освоєння мови і, головним чином, саме на процес написання ПЗ цією мовою.

Універсальність визначається можливостями мови для опису різноманітних алгоритмів, характерних для програмного забезпечення конкретного проекту САПР.

Ефективність об’єктних програм (тобто програм, отриманих після трансляції на машинну мову) оцінюється витратами машинного часу і пам’яті на їх використання в процесі експлуатації САПР.

Машино-орієнтовані мови (мови асемблера або автокоди) найбільш відповідають вимогам універсальності і ефективності об’єктних програм. Ці мови найбільш близькі до мов машинних команд і тому для їх перекладу потрібно прості транслятори (асемблери). Проте мови асемблера незручні для людини. Їх використовують для розробки тільки тих модулів в САПР, які вимагають для роботи великих обчислювальних ресурсів та істотно впливають на загальні витрати часу і пам’яті в процесі експлуатації.

Мови проектування призначені для опису інформації про об’єкти і задачі проектування.

Вхідні мови - служать для подання початкової інформації про об’єкти і задачі проектування і включають мови опису об’єктів (МОО) і мови опису задач (МОЗ). Перші служать для опису властивостей об’єктів проектування, другі - для опису задач на виконання проектних процедур.

Схемні мови застосовують для опису принципових електричних схем при проектуванні електронних пристроїв.

Графічні мови - основа лінгвістичного опису об’єктів геометричного моделювання і машинної графіки.

Мови моделювання - використовують для опису інформації, наданої алгоритмом функціонування ОП. Наприклад, імітаційного моделювання систем масового обслуговування.

Вихідні мови використовують для вираження результатів виконання проектних процедур на ЕОМ.

Мови супроводу застосовують для коригування і редагування даних при виконанні проектних процедур в процесі експлуатації САПР.

Проміжні і внутрішні мови.

Проміжна мова призначена для представлення інформації на певних стадіях її обробки на ЕОМ. Проміжні мови є більше універсальними, ніж вхідні мови, для яких характерна вузька проблемна орієнтація. Вони відображають особливості широкого класу об’єктів проектування і в певному значенні є інваріантними.

Проміжні мови використовують для організації так званих програмних систем дворівневого лінгвістичного забезпечення.

Конвертор - це спеціальна транслююча програма, яка перекладає опис з вхідної мови на проміжну.

Переваги дворівневого лінгвістичного забезпечення полягають в тому, що програмна система легко налаштовується

на нові класи об’єктів та не вимагає значних машинних ресурсів в процесі експлуатації.

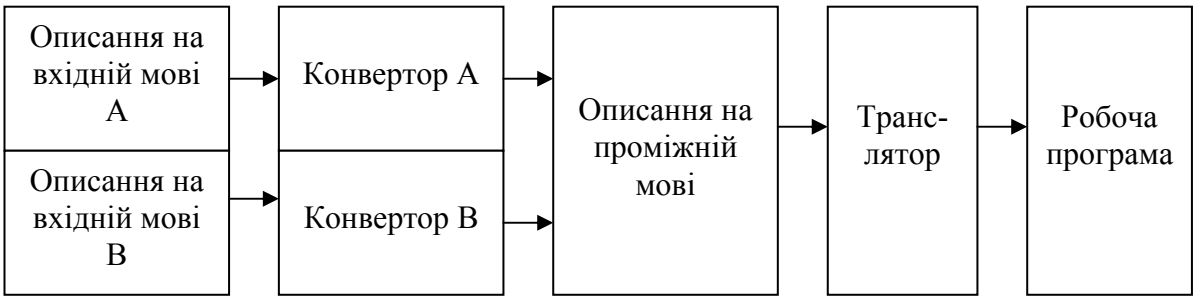


Рисунок 2 – Структура програмної системи САПР

1.2 Діалогові мови

Діалогові мови призначені для організації і ведення діалогового режиму в процесі функціонування САПР. Вони об’єднують в собі засоби вхідного введення інформації про об’єкт проектування, одержання вихідних результатів і супроводу ПЗ в процесі експлуатації САПР та служать для оперативного обміну інформацією між людиною і ЕОМ.

Розрізняють пасивні і активні діалогові мови, які використовують для організації, відповідно, пасивного і активного діалогових режимів.

У пасивному діалоговому режимі ініціатива діалогу належить ЕОМ. У заздалегідь визначених точках виконання програми передбачається можливість переривання обчислювального процесу і звертання системи до користувача.

Повідомлення системи будуються таким чином, що від користувача потрібно відповіді типу «та або ні» або вибір відповіді з наданого ЕОМ меню. (Приклади з AUTOCAD).

Тому мова користувача виявляється дуже простою - вона складається з дій, що означають «так», «ні» або є підтвердженням/вибором з безлічі варіантів відповідей.

Для використання пасивних мов практично не потрібна будь-яка спеціальна підготовка користувача в області лінгвістичного забезпечення САПР, оскільки пасивні мови використовують типи звернень на зрозумілій для користувача мові.

Розрізняють наступні види (типи) звернень: запит; інформаційне повідомлення; підказка.

Запит передбачений в двох випадках: коли від користувача вимагаються початкові дані; коли вимагається виконати вибір з безлічі можливих пропозицій проектування. При запиті варіанту відповіді користувачу зазвичай пропонується «меню».

У багатьох існуючих САПР організований вибір «за умовчанням», тобто - автоматичний вибір деякого основного або поточного варіанту, коли, наприклад, користувач не може надати певну відповідь на поставлене питання.

Інформаційне повідомлення використовується для передачі користувачу проміжних і остаточних результатів одержаного проектного рішення, а також відомостей про стан його задачі. Ці повідомлення не вимагають реакції користувача.

Підказка застосовується в тих випадках, коли дії користувача помилкові, наприклад, при граматичних помилках користувача або при виборі варіантів з «меню» і т. п. Зазвичай, на екран монітора виводиться повідомлення про допущену

користувачем помилку та пропонується звернутись до джерела інформації, яке дозволить одержати відповідну інформацію для виправлення помилкових дій.

1.3 Організація діалогу в САПР

Діалогова взаємодія (діалог) є регламентованим обміном інформацією між людиною і обчислювальною машиною, здійснюється в реальному масштабі часу і спрямовується на спільне рішення проектної задачі.

Ініціатором діалогу є людина, яка вибирає мету і в якійсь мірі може впливати на способи її досягнення. Розподіл функцій між людиною і ЕОМ полягає в наступному: людина ставить завдання; система представляє засоби для вирішення підзадач; виробляється спільне рішення підзадач, яке дозволяє остаточно об’єднати результати; ухвалення проектних рішень залишається за людиною.

Поняття діалогової системи (ДС).

Комплекс засобів автоматизації в САПР є партнером людини по діалогу і представлений у вигляді діалогової системи.

Діалоговою системою (підсистемою) називають систему (підсистему), яка забезпечує функціонування в режимі діалогу. ДС можуть бути розрахованими на одного користувача, на багато користувачів з колективним доступом до ресурсів системи.

Технічною базою діалогової системи можуть бути: локальні і віддалені термінали у складі центрального

обчислювального комплексу; АРМ; локальна мережа АРМ; мережа ПК, що включає ЦВК, АРМ.

Поняття повідомлення.

Обмін інформацією між партнерами діалогу здійснюється за допомогою передачі повідомлень і керуючих сигналів. До складу повідомлення входить інформація наступного виду: пояснення; попередження; навчання; вказівки тощо. Серед безлічі діалогових повідомлень розрізняють вхідні і вихідні.

Вхідні повідомлення виробляються людиною за допомогою засобів введення. Вихідні повідомлення формуються системою на екрані терміналу у вигляді тексту, таблиць, графіків або зображення.

1.4 Діалогові обміни

Діалоговий обмін - це елементарний крок (квант, частина) діалогу, який включає наступні фази: видача вихідного повідомлення; аналіз повідомлення користувачем; введення вхідного повідомлення в ПК; виконання обробки введеної інформації, як це показано на рисунку 3.

Діалоговий обмін зв’язує вихідні, вхідні повідомлення та програми обробки. Основним компонентом діалогу є розмова. Це інформаційно пов’язана послідовність діалогових обмінів, спрямована на виконання функцій системи.

Не менш важливим компонентом діалогу являється діалогова процедура, яка включає окрім людино-машинних розмов і діалогових обмінів також ручні операції користувача і машинні процедури, як це показано на рисунку 4.

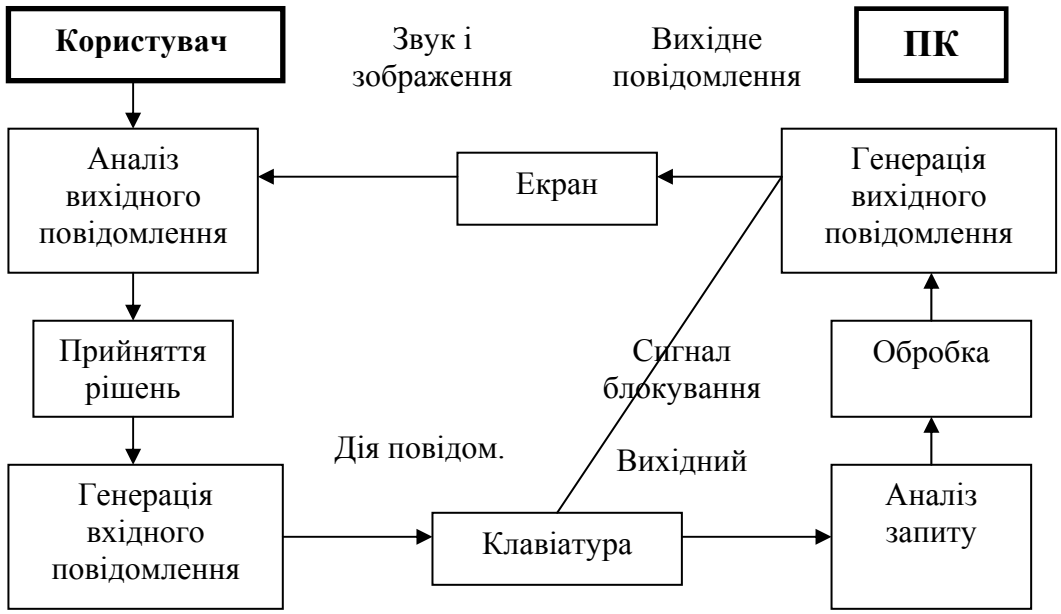


Рисунок 3 - Структурна схема діалогової взаємодії людини і ПК

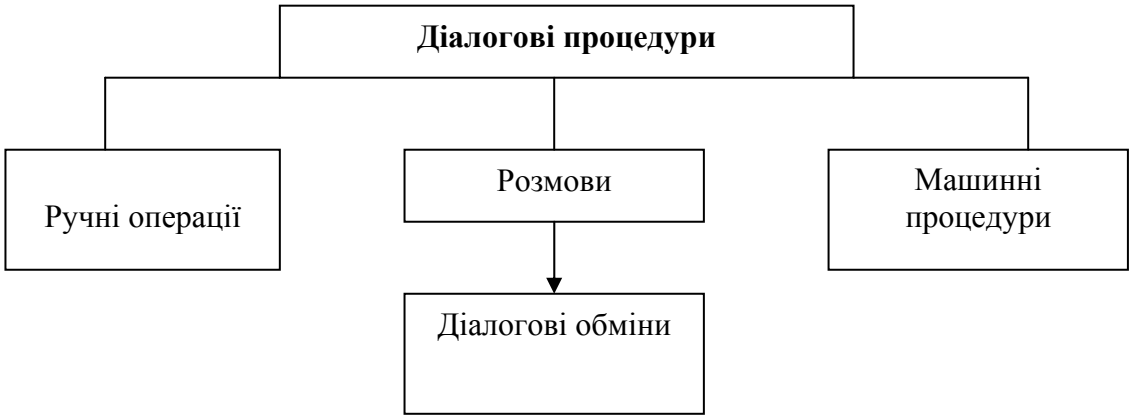


Рисунок 4 - Ієрархія елементів діалогу

1.5 Способи взаємодії людини і комп'ютера

При організації діалогу можлива синхронна і асинхронна взаємодія людини і ПК.

Синхронний спосіб взаємодії характеризується тим, що партнери діалогу активізуються в будь-який термін часу (поза чергою) в разі виникнення потреби.

Асинхронний спосіб взаємодії забезпечує: можливість видачі екстрених повідомлень від системи, які переривають процес набору вхідного повідомлення; введення екстрених запитів користувача, який може призупинити виведення повідомлень системи.

Варіанти асинхронного діалогу :

а) двофазна обробка запитів;

б) виведення системи з оперативним втручанням користувача із нештатної ситуації.

У активному діалоговому режимі ініціатива початку діалогу може бути двосторонньою, тобто можливості переривання обчислювального процесу належить як комп'ютеру, так і користувачу.

Користувач може в будь-який момент перервати обчислення і звернутися до комп'ютера.

Активні діалогові мови близькі до природної мови людини але з обмеженим набором можливих слів і фраз. В той же час, число різних директив, тобто приписів для обчислювальної системи, може бути порівняно великим.

Для активного діалогу потрібне істотно складніше ПЗ, ніж для пасивного.

2 Програмне забезпечення САПР

ПЗ займає особливе місце в САПР, оскільки саме програмними засобами реалізуються методи автоматизованого проектування. Складність ПЗ пояснює великі витрати коштів

на його розробку - до 90% від загальної суми, що виділяється на створення САПР.

2.1 Склад ПЗ

ПЗ САПР є сукупністю програм на машинних носіях, призначеною для виконання автоматизованого проектування. Воно підрозділяється на базове, загальносистемне і спеціалізоване.

Технічні засоби САПР (ТЗ) працюють в середовищі цих видів ПЗ, які взаємодіють в процесі роботи (рисунок 5).

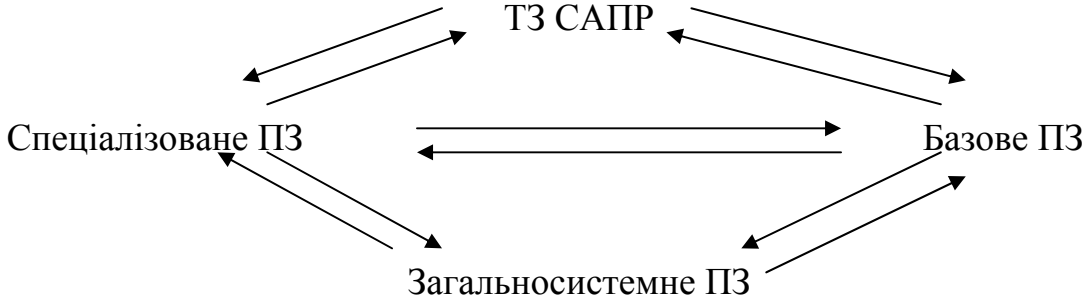


Рисунок 5 – Структурна схема взаємодії ПЗ САПР з технічними засобами САПР

Базове і загальносистемне ПЗ утворює операційне середовище, тобто операційну систему, в якій функціонує спеціалізоване ПЗ:

$$Б_{ПЗ} + ОС_{ПЗ} = ОС$$

Функція спеціалізованого ПЗ - отримання проектних рішень.

Операційні системи включають програми двох груп: обробники, управляючі (рисунок 6).

Програми управління задачею виконуються за допомогою мови управління задачею. Наприклад, за допомогою цієї мови можна задати машині наступну послідовність дій: введення; трансляція; завантаження в пам’ять машини даних; виробка рішення; виведення інформації.

Програми управління даними забезпечують пошук, зберігання, завантаження програм в ПК, обробку файлів.

Програми обробки - це транслятори з алгоритмічними мовами, бібліотеками стандартних програм і системним обслуговуванням сервісних програм.

Програми користувача на алгоритмічній мові - це початкові модулі.

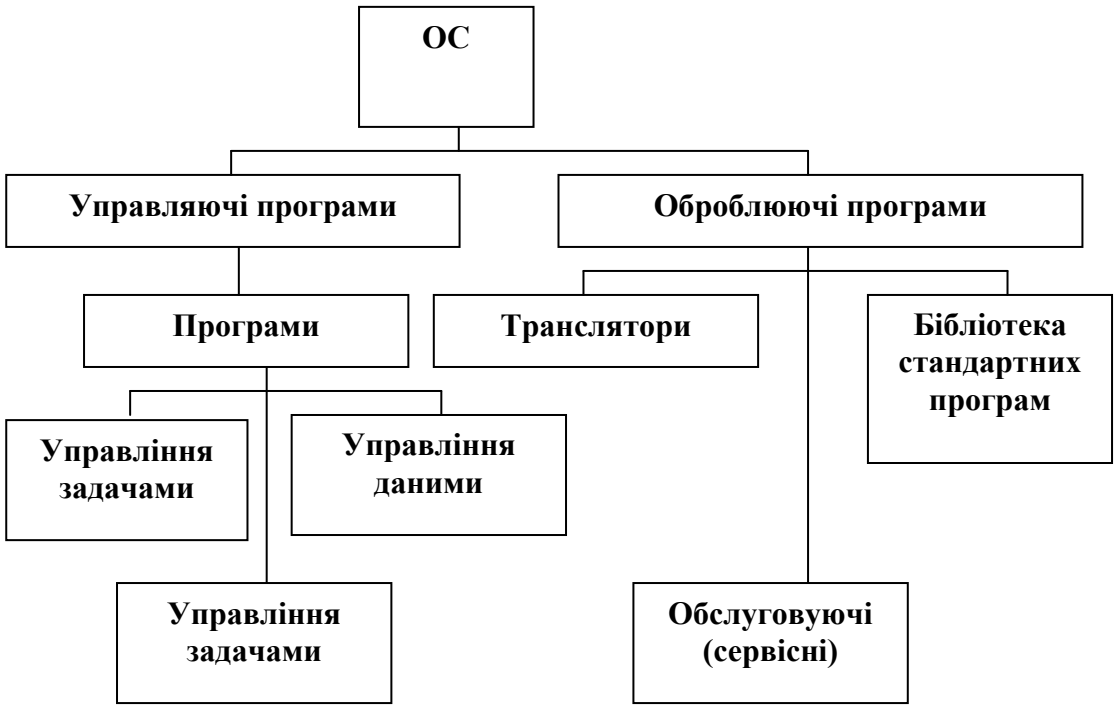


Рисунок 6 – Структурна схема взаємодії видів ПЗ

В результаті трансляції отримують програму, яка називається об’єктним модулем. Різні об’єктні модулі збираються в єдину програму за допомогою обслуговуючої програми, яку називають редактором зв’язків або просто редактором. Результат редагування - це програма що називається завантажувальним модулем. Програма-завантажувач виконує редагування і завантаження програм в ПЗ.

Приклади інших оброблювальних програм: програма - відладчик для налагодження програм, тобто для забезпечення і прискорення пошуку допущених помилок.

2.2 Класифікація ПЗ САПР по функціональному призначенню

По функціональному призначенню ПЗ САПР розподіляються на ряд програмних комплексів або підсистем. Можна виділити наступні види підсистем ПЗ САПР: проектувальні, обслуговуючі і інструментальні (рисунок 7).

Рисунок 7 - Класифікація ПО САПР по функціональному призначенню

Проектувальні підсистеми призначені для отримання закінченого проектного рішення і діляться на проблемно-орієнтовані і об’єктно-орієнтовані.

Проблемно-орієнтовані підсистеми виконують уніфіковані проектні процедури, не залежні від об’єкту проектування.

Об’єктно-орієнтовані підсистеми використовуються для проектування об’єктів певного класу і входять до складу спеціалізованого ПЗ.

Обслуговуючі підсистеми призначені для підтримки працездатності підсистем, які входять до складу загальносистемного ПЗ.

Інструментальні підсистеми - це технологічні засоби, призначені для розробки, розвитку і модернізації ПЗ САПР. До складу інструментальних засобів, які використовують в процесі роботи САПР входять: системи управління базами даних (СУБД) і файлами; засоби для роботи із загальними структурами даних; мовні процесори для забезпечення взаємодії з користувачами (діалогові); засоби машинної графіки.

2.3 Основні принципи проектування ПЗ САПР

Принцип системної єдності означає, що окремі компоненти єдиного програмного комплексу ПЗ САПР повинні забезпечувати його цілісність.

Принцип розвитку - ПЗ САПР повинно створюватися і функціонувати з урахуванням поповнення, вдосконалення і

оновлення, при цьому не вимагаючи додаткових матеріальних витрат.

Принцип сумісності - мови, символи, коди, інформація і зв'язки між ними повинні забезпечувати їх спільне функціонування і зберегти відкриту структуру системи в цілому.

Принцип стандартизації. При розробці ПЗ САПР необхідно уніфікувати і стандартизувати ПЗ, інваріантне з проектованим об'єктом.

Загальні вимоги, що пред'являються до ПЗ САПР відповідні загальним принципам створення САПР. До них віднесемо наступні:

- адаптація - пристосованість ПЗ до функціонування в різних умовах. Це пов'язано зі зміною самих об'єктів проектування;

- гнучкість - можливість легко вводити зміни, доповнення чи заміну модулів наявного ПЗ при збереженні усієї системної організації;

- компактність - споживання мінімальних ресурсів ЕОМ (пам'яті, часу центрального процесора ЕОМ);

- мобільність - здатність функціонування ПЗ САПР на різних ТЗ;

- надійність - забезпечення - отримання достовірних результатів проектування;

- реактивність - забезпечення швидкого рішення задачі при орієнтації на користувача, що не є фахівцем в області ЗОТ і програмування;

- еволюційність - поповнення САПР новими програмами, що розширюють можливості системи.

2.4 Модульний принцип побудови програм

Спеціалізоване ПЗ - це складний комплекс програм, що налічує десятки, сотні, тисяч операторів, алгоритмів та декілька мов програмування. Для успішного створення такого складного програмного комплексу його розділяють на модулі, які є до певної міри самостійними програмними компонентами.

До переваг модульного принципу побудови САПР віднесемо наступне - міра автономності модулів повинна забезпечувати їх розробку незалежно один від одного. Тоді програмування модулів по сформованому ТЗ виконують паралельно в часі декілька програмістів. Проте модулі не мають бути занадто дрібними, оскільки зайве дроблення, тобто ускладнення структури, призведе до збільшення числа міжмодульних зв’язків.

Модульна побудова спеціалізованого ПЗ робить чіткою і зрозумілою його структуру. Це зменшує число помилок, що допускаються при програмуванні, і спрощує налагодження програм.

У різних маршрутах проектування має місце велика кількість близьких за змістом операцій, які можуть бути реалізовані по типових програмах. Тому при модульній побудові спеціалізованого ПЗ створюють обмежене число як типових, так і нетипових модулів. Різний зміст таких модулів забезпечує велике число маршрутів проектування, тобто будь-

який новий, заздалегідь непередбачений маршрут, вдається реалізувати на основі вже наявних модулів або взагалі без їх доробки.

Модульна структура спеціалізованого ПЗ має властивість ієрархічності. Це означає, що будь-яку сукупність модулів доцільно вважати окремим модулем. Для цієї вказівки сукупність повинна мати ознаки самостійної програми або входити, як складова частина, в декілька поєднань модулів, що реалізовують маршрути проектування.

3. Інформаційне забезпечення САПР

Призначення інформаційного забезпечення **(**ІЗ**)** - надання користувачам САПР необхідних даних для виконання передбачених в САПР проектних операцій і процедур. Дані – це відомості про деякі факти, що дозволяють робити певні висновки. Іншими словами, основне призначення ІЗ полягає в представленні користувачам САПР достовірної інформації в певному вигляді.

Приклади ІЗ САПР: типові проектні рішення (наприклад, опис технологій); типові елементи і комплектуючі вироби; каталоги (верстати, інструменти, оснащення, пристосування тощо).

3.1 Банки даних (БнД)

Розрізняють два основні види інформаційних систем: банки даних (БнД), інформаційно-пошукові системи (ІПС). Ці

системи розрізняються в основному інформаційною мовою, за допомогою якої здійснюється опис інформації і маніпуляції над нею.

Банки даних складаються з бази даних (БД) і системи управління базою даних, тобто:

$$БнД = БД + СУБД.$$

Інформаційна мова БнД - це сукупність двох мов: мови опису структури даних (ОД); мови маніпулювання даними (МД).

БнД - сукупність спеціально організованих даних, розрахованих на застосування у великій кількості застосовних програм, тобто тих даних, які обробляються більш, ніж в одній програмі (програмному модулі). В БД дані мають бути представлені в одній з форм, допустимих в даному БнД.

СУБД реалізується за допомогою спеціального ППП, який призначений для: накопичення (введення; зміни; модифікація); зберігання; пошуку інформації (витягання). Тобто, СУБД реалізує доступ до БнД.

Види даних, що зберігаються у БД:

- архів - довідкові дані про типи, параметри, структуру уніфікованих деталей і приладів; типових проектах і технологічних процесах, матеріалах і так далі;

- робочий масив (РМ) - містить результати виконання попередніх етапів проектування конкретних об’єктів. Призначений для використання на подальших етапах. Складові РМ представлені у вигляді кодів, записаних на різних програмних накопичувачах. Приклади РМ: конструкторські

документи, технічні описи, технологічні процеси, тощо. РМ можна називати інформацією про проект.

3.2 Інформаційні потоки в САПР.

Інформаційні потоки в САПР зручно показувати у вигляді обміну даними (рисунок 8).

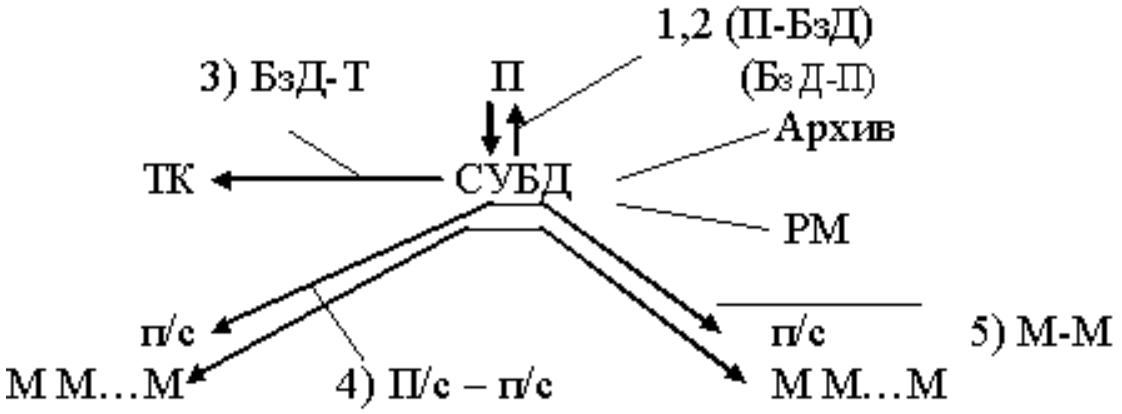


Рисунок 8- Структура інформаційних потоків в САПР: П - користувач; ТК - технологічний комплекс; п/з - підсистема САПР; М - модуль (програмний). Потоки: П – БзД, БзД – П, БзД – ТК, п/з - п/з, М – М.

Потоки 1 і 2 використовуються при занесенні або коригуванні даних у БзД (БД) інженерами користувачами САПР або отримана ними необхідна інформація з БзД.

Потік 3 потрібний для передачі управляючої інформації, яка отримується в САПР, на автоматичне технологічне устаткування.

Потік 5 служить для інформаційного зв’язку програм, що входять в маршрут проектування, між собою, а потік 4 - для інформаційного зв’язку інших підсистем. САПР Наявність

потоків ПС - ПС(4) - необхідна умова реалізації наскрізного автоматизованого проектування складних об’єктів.

3.3 Функціональний розподіл БзД

У великих САПР БнД зазвичай складається з декількох частин: центрального банку (ЦБнД) і локальних банків даних (ЛБнД). ЦБнД містить відомості, використовувані різними СУБД, яка в цьому випадку є спільно цільовою. ЛБнД прив’язаний до конкретної САПР і має спеціалізовану СУБД (рисунок 9).

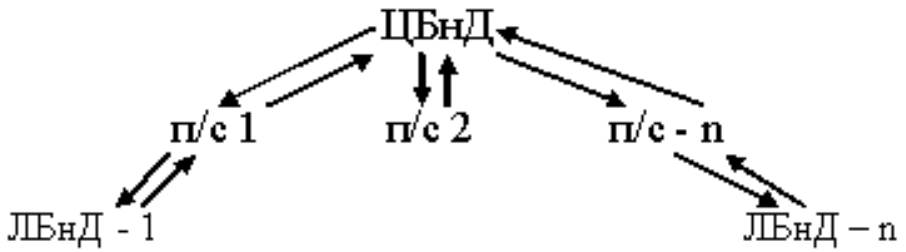


Рисунок 9 - Вимоги до БзД

Зазвичай, при формуванні БзД необхідно забезпечити наступні вимоги:

- Повнота інформації - означає достатність даних з нормативних документів і отриманих в результаті виконання попередніх етапів для виконання проектної процедури.

- Ненадмірність інформації (скорочення надмірності). Одні і ті ж відомості використаються при розробці різних САПР, входять в масиви початкових даних для багатьох програм. Це може призвести до багатократного дублювання одних і тих же даних в різних частинах БзД. Відсутність такого

дублювання означає виконання принципу ненадмірності, що дозволяє уникнути нераціонального використання пам'яті.

- Достовірність інформації - означає відсутність суперечливих, невірних, надмірних даних у БзД. Для підтримки достовірності інформації необхідно своєчасно вносити зміни, причому одночасно повинні мінятися дані в усіх масивах, в яких є дублювання відомостей. Достовірність забезпечується також системою захисту даних від випадкових накладень і змін, а саме:

- Незалежність представлення даних від типів запам'ятовуючих пристроїв і способів їх фізичної організації.

- Швидкодія.

- Мінімізація витрат пам'яті.

- Захищеність даних.

Це досягається дворівневою системою представлення інформації (даних): логічного і фізичного рівнів. Логічний рівень - містить уявлення про структуру даних і призначений для використання розробниками програм. Фізичний рівень - відбиває спосіб зберігання і структуру даних з урахуванням їх розміщення на конкретних носіях інформації в запам'ятовуючих пристроях ЕОМ.

Швидкодія пов'язана з необхідністю мінімізації витрат часу на доступ до інформації. Наприклад, при зверненні до БзД в діалоговому (інтерактивному) режимі, реакція системи не повинна затягуватися більш, ніж на декілька секунд.

Мінімізація витрат пам'яті. Ця вимога виконується шляхом усунення надмірності даних, своєчасного видалення

непотрібних даних, скиданням застарілих відомостей на носії, призначенні для довгострокового зберігання інформації.

Захищеність даних означає властивість системи протистояти несанкціонованому доступу до даних. Захист реалізується в самій СУБД, яка порівнює інформацію користувача, що зберігається в системі, з шифром, вказаним при вході в систему.

3.4 Структура БД (моделі даних)

Існують три основні структури БД (в залежності від моделі представлення даних): реляційна (РМД); ієрархічна (ІМД); мережева (ММД). Розглянемо кожну з них.

Реляційна модель даних (реляційна структура) - є простими двовимірними таблицями. Кожна таблиця містить групу однотипних записів, причому кожен запис може бути пов’язаний з іншими таблицями. БД, ті, що мають подібну структуру називаються реляційними (рисунок 10).

Кожен стовпець в таблиці називається атрибутом. Рядки є кортежами, тобто впорядкованими множинами. Стовпці в таблиці - елементи даних, рядки - записи.

Запис - це об’єднання значень пов’язаних атрибутів. Атрибут - деяка характеристика об’єкту (швидкість, потужність). Впорядкована сукупність записів даних називається файлом даних або набором даних.

Ідентифікаційний номер.	Прізвище
246	Іванов
344	Петров
…	…
422	Сидоров

Рисунок 10 – Приклад структури реляційної таблиці

Переваги реляційної моделі: простота; незалежність даних; гнучкість.

Ієрархічна модель даних має деревовидну структуру (рисунок 11).

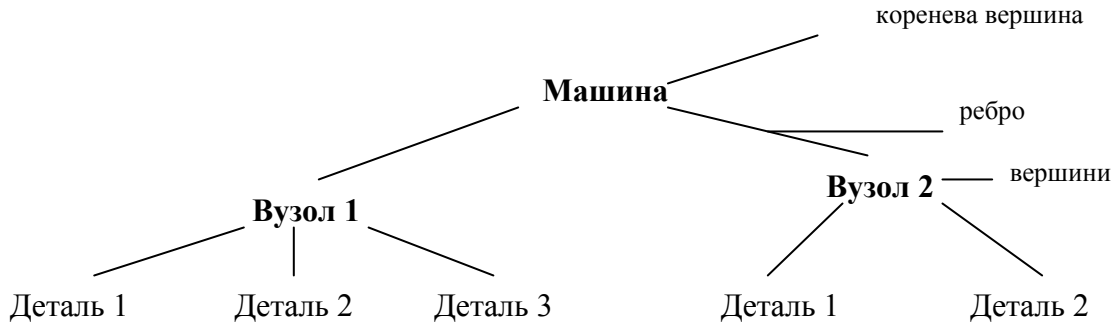


Рисунок 11 - Ієрархічна деревовидна структура БзД

Вершина дерева ставиться у відповідність сукупності атрибутів даних, що характеризують деякий об’єкт проектування, а ребра відбивають сенсові зв’язки. Наводимо типи запитів до ієрархічних БД: знайти вузли для заданих пристроїв (машин), знайти деталі для заданих вузлів тощо.

Труднощі при використанні ієрархічної БД виникають при зміні типу запитів. Наприклад, якщо з’явиться запит типу «Знайти вузли, що включають задану деталь», то необхідно

продивитися усі записи вузлів і усі пов’язані з ними записи деталей, що незручно. Для реалізації такого запиту було б доцільно переупорядкувати БзД і побудувати нову ієрархію від записів про деталі до записів про вузли або використати мережеві моделі.

Мережеві моделі даних розробляють з метою задоволення різних типів запитів з високою швидкодією. В мережевій моделі даних дозволені будь-які групування записів і організація довільних зв’язків між ними (рисунок 12).

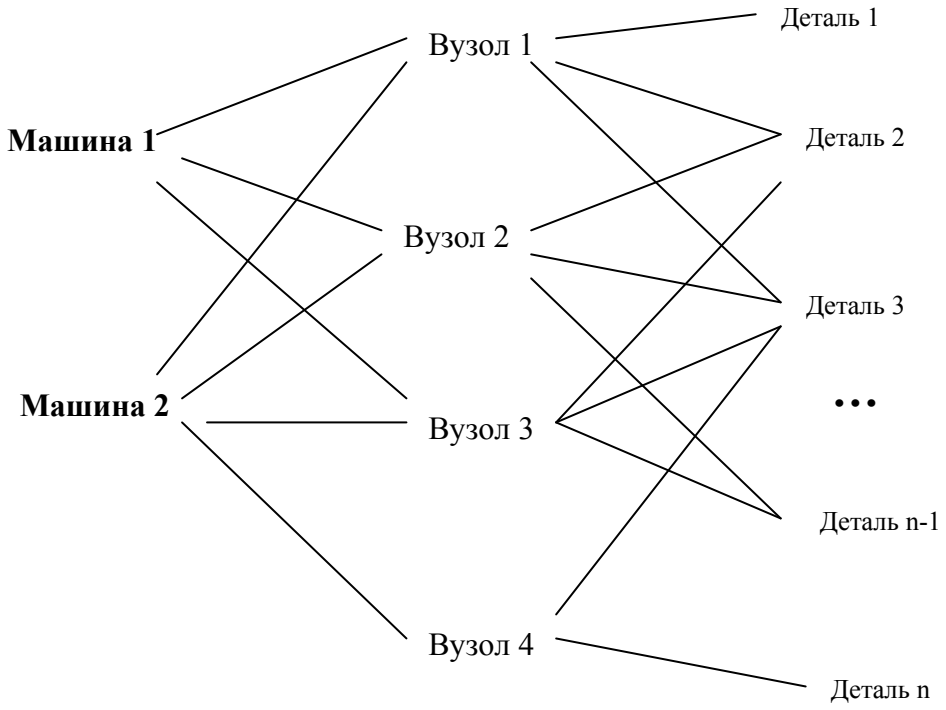


Рисунок 12 – Структура мережевої моделі даних

4 Технічне забезпечення САПР

Технічне забезпечення САПР спільно з ПЗ є інструментальною базою САПР, в середовищі якої реалізуються інші види забезпечень САПР (рисунок 13).

Компоненти технічного забезпечення САПР: засоби ОТ – засоби обчислювальної техніки; оргтехніка; засоби передачі даних; вимірювальні та ін. пристрої (рисунок 14).

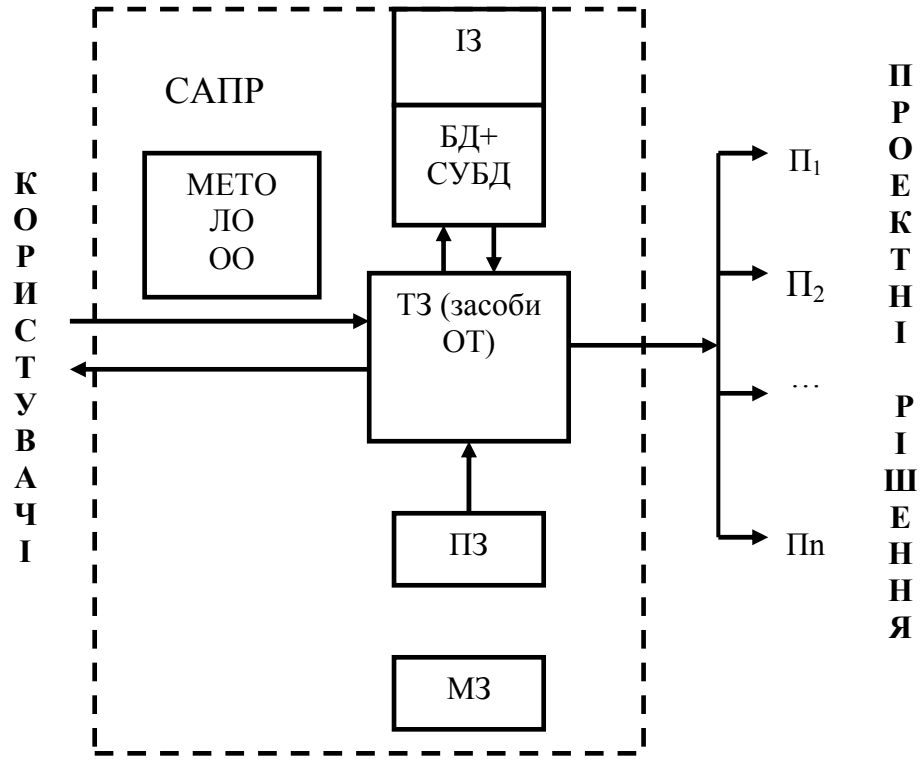


Рисунок 13 - Роль і місце технічного забезпечення в структурі САПР

Функції технічного забезпечення САПР (вирішувані завдання):

- введення початкових даних опису об’єкту проектування (введення зображення за допомогою пристроїв введення/виведення (клавіатура, миша);

- відображення введеної інформації з метою її контролю і редагування; (отримання зображення на екрані);

- перетворення інформації(операції по редагуванню);

- відображення підсумкових і проміжних результатів рішення (згадаємо команди MOVE, COPY, MIRROR) - є

присутнім після позначення об’єкту і задання базової точки користувачем, проміжне зображення, потім після команди ENTER остаточне;

- документування проектної інформації (креслення, оформлені відповідно до ЄСКД);

- оперативне спілкування проектувальника з системою в процесі рішення задачі (це те, що ми раніше називали діалогом людини і машини - машина реагує на запити користувача, користувач відповідає на запити машини та може призупинити обчислювальний процес або діалог командою Ctrl – C).

Для виконання рішення цих задач (функцій) технічне забезпечення САПР повинне мати у своєму складі: процесори, оперативну пам’ять (ОП), зовнішні пристрої (ВЗП), пристрої введення/виведення інформації (ПВВІ), технічні засоби машинної графіки, пристрої оперативного спілкування людини з ЕОМ, пристрої, що забезпечують зв’язок ЕОМ з віддаленими терміналами та ін. машинами (рисунок 14).

При необхідності створення деякого родинного зв’язку САПР з технологічним устаткуванням до складу ТЕ мають бути включені пристрої, що перетворюють результати проектування в сигнали управління верстатами, різними технологічними комплексами.

Розглянуті функції технічного забезпечення вирішуються спільно із загальносистемним ПЗ (тобто операційними системами ЕОМ).

- ВЗП – зовнішні запам’ятовуючі пристрої;

- ПВВИ – пристрої введення - виведення інформації;

- ПОС - пристрої облаштування оперативного зв’язку;

- ПМГ - пристрої машинної графіки ;

- ППД - пристрої підготовки даних;

- НМД, НГМД, НМЛ - накопичувачі на магнітних дисках, на гнучких магнітних дисках.

До центральних пристроїв, що здійснюють безпосередню обробку даних, відносяться центральний процесор, ОЗУ і процесор введення/виведення.

До периферійних пристроїв відносяться пристрої, які виконують функції введення/виведення, підготовки даних і зберігання великих об’ємів інформації.

Центральний процесор призначений для перетворення інформації відповідно до виконуваної програми, управління обчислювальним процесом і пристроями, працюючими спільно з процесором.

ОЗУ - виконує функції зберігання, прийому і видачі даних і програм.

Процесори введення/виведення призначені для управління обміном інформацією між ОЗУ і периферійними пристроями без участі центрального процесора; узгодження швидкості роботи ПП і ОЗУ, уніфікації програмування, введення/виведення і забезпечення можливості підключення нових ПП.

Технічні засоби теледоступу і мереж ЕОМ здійснюють зв’язок ЕОМ з віддаленими користувачами САПР, між усіма ЕОМ, що входять в КТЗ САПР при організації мережі проектування.

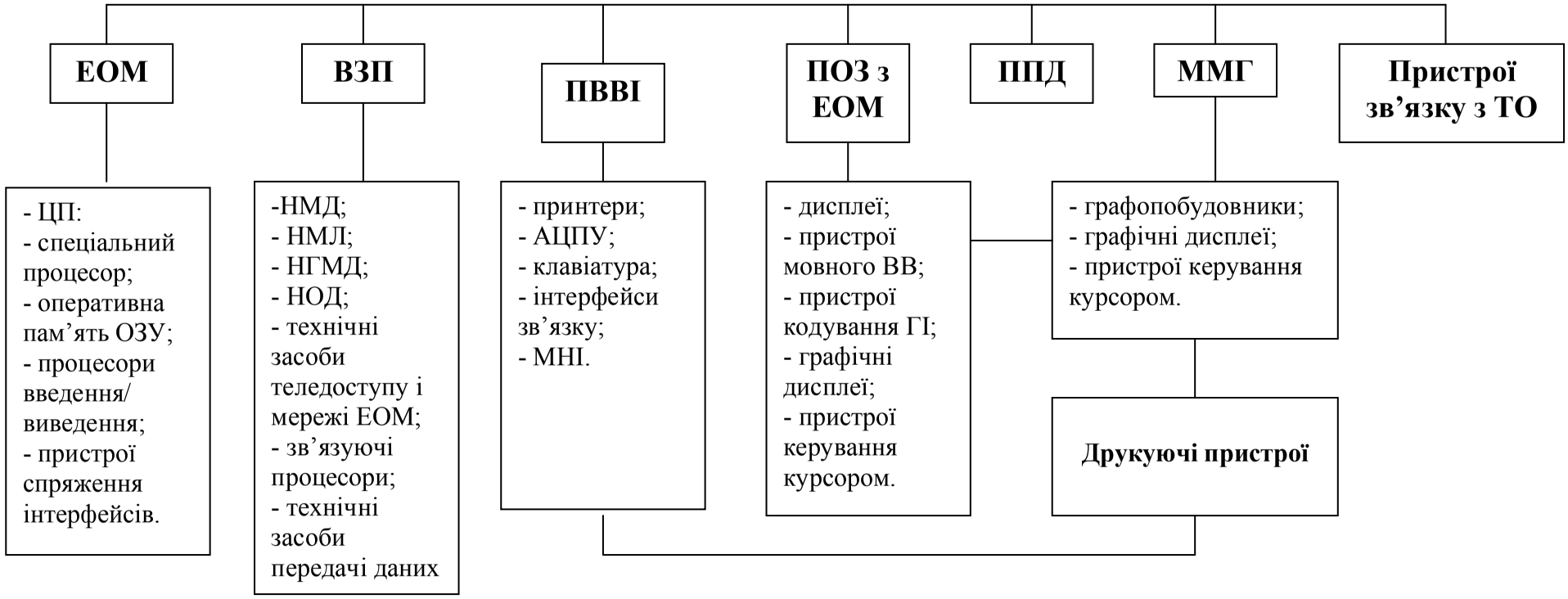


Рисунок 14 - Склад технічного забезпечення САПР

.

Вимоги до технічних засобів і ОС САПР:

- забезпечення комфортних умов проектувальнику (нові клавіатури, столи, стільці, захисні екрани);

- достатність параметрів ЕОМ для вирішення усіх проектних задач;

- можливість оперативної взаємодії проектувальника з ЕОМ в процесі проектування;

- прийнятний для проектувальника час реакції системи на його запити;

- простота освоєння експлуатації і обслуговування технічних засобів;

- можливість графічного представлення інформації.

МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ САПР. МАТЕМАТИЧНЕ МОДЕЛЮВАННЯ В САПР

1 Структура математичного забезпечення САПР

Математичне забезпечення - це сукупність математичних моделей, методів, алгоритмів для вирішення задач автоматизованого проектування.

Елементи математичного забезпечення різноманітні. Це - і принципи побудови функціональних моделей, методи чисельного рішення рівнянь алгебри і диференційних рівнянь, постановка екстремальних завдань, пошук екстремуму. Математичне забезпечення реалізується в програмному забезпеченні САПР.

Математичне забезпечення складається з трьох великих блоків: математичні моделі; чисельні методи; алгоритми.

2 Математичні моделі

Математичні моделі (ММ) служать для опису властивостей об’єктів в процедурах САПР. До математичних моделей пред’являються вимоги: універсальності, адекватності, точності, економічності.

Міра універсальності ММ характеризує повноту відображення в моделі властивостей реального об’єкту.

Точність ММ оцінюється мірою збігу значень параметрів реального об’єкту і значень тих же параметрів, розрахованих за допомогою оцінюваної моделі.

Адекватність ММ - здатність відображати задані властивості об’єкту з погрішністю не вище за задану.

Економічність ММ характеризується витратами обчислювальних ресурсів на її реалізацію.

Виходячи з властивостей ОП, ММ, діляться на: структурні, функціональні

Структурні ММ призначені для відображення структурних властивостей об’єкту і підрозділяються на: топологічні, геометричні. У топологічних ММ відображаються склад і взаємозв’язки елементів ОП. У геометричних ММ відображаються геометричні властивості ОП.

Функціональні ММ призначені для відображення фізичних або інформаційних процесів, що протікають в ОП при його функціонуванні або виготовленні. Зазвичай функціональні

ММ є системами рівнянь, що зв’язують фазові змінні, внутрішні, зовнішні і вихідні параметри.

Функціональні ММ діляться на: аналітичні, - алгоритмічні.

Використання принципів блоково-ієрархічного підходу до проектування призводить до появи ієрархії математичних моделей проектованих об’єктів.

Математичні моделі підрозділяються на мікромоделі і макромоделі. Мікромоделі - цей якнайповніший і точніший опис технічного об’єкту. Особливістю ММ на мікрорівні є віддзеркалення фізичних процесів, що протікають у безперервному просторі і часі. Основою для отримання мікромоделей є фундаментальні науки: фізика, хімія, фізична хімія. Типові ММ на мікрорівні - диференціальні рівняння в приватних похідних і інтегро-дифференційні рівняння.

Мікромоделі мають високу міру універсальності, оскільки враховують усі властивості об’єкту, але застосування їх в САПР практично неможливе, оскільки витрати машинного часу дуже великі. Тому в САПР широко застосовують макромоделі об’єктів.

Макромоделі - це моделі, в яких враховуються властивості, істотні для вирішуваної задачі. На макрорівні використовують укрупнену дискретизацію виробництва за функціональною ознакою, що призводить до представлення ММ на цьому рівні у вигляді систем звичайних рівнянь алгебри і диференційних рівнянь .

Методи отримання макромоделей діляться на теоретичні і емпіричні. Теоретичні моделі отримують шляхом зведення мікромоделей до макромоделей. Емпіричні моделі отримують, досліджуючи відгук об’єкту на різні зовнішні дії, а потім апроксимуючи отримані залежності на безлічі заданих функцій. Макромодель може бути описана таблицями.

Для отримання чисельних рішень, математичні моделі, виражені системами рівнянь, мають бути зведені до елементарних операцій, доступних для виконання на ЕОМ. Для отримання рішення рівнянь, що описують математичні моделі, використовують чисельні методи.

2.1 Чисельні методи рішення рівнянь, обчислень, пошук екстремумів

Чисельні методи є одним з розділів прикладної математики. Це - методи наближеного рішення математичних завдань шляхом зведення їх до виконання кінцевого числа арифметичних дій над числами. У чисельних методах досліджується питання про точність отримуваних рішень і надійності методів їх отримання. Для одного і того ж завдання, наприклад, рішення нелінійного рівняння, існує декілька методів: ділення відрізку навпіл, золотого перерізу, Ньютона - Рафсона. Обґрунтування вибору ефективного методу для вирішення конкретної системи рівнянь або розрахунку функції і є основна проблема чисельних методів.

Системи диференціальних рівнянь в приватних похідних і системи звичайних диференціальних рівнянь вирішуються

шляхом зведення їх до систем рівнянь алгебри (нелінійних) і систем лінійних рівнянь алгебри. Крім того, значна частина завдань зводиться до рівнянь алгебри, тому велику роль грають методи рішення цього класу рівнянь. Однією з поширених груп методів рішення рівнянь алгебри є методи ітерації.

Окрім цієї групи методів існують методи рішення лінійних рівнянь, що враховують їх особливість: метод Гауса, метод ВН - матриць і цілий ряд інших. Велика увага до рішення систем лінійних рівнянь алгебри пояснюється тим, що багато завдань рішення рівнянь зводяться до таких систем.

Окрім рішення систем рівнянь чисельними методами, розроблені способи обчислення функцій, інтегралів, похідних, які дозволяють виконувати розрахунки із заданою точністю з найменшими витратами ресурсів. Широке використання в інженерно - технічних розрахунках отримали методи інтерполяції і екстраполяції, що дозволяють по деяких відомих точках функції отримати її значення або між цими точками або поза областю їх визначення.

Для вирішення завдання оптимізації необхідно уміти знаходити екстремум цільової функції. Рішення цих завдань також добре розвинені в чисельних методах. Особливо широке застосування отримали методи, засновані на обчисленні градієнта функції. На основі виводів чисельних методів створюють алгоритми рішення математичних задач.

2.2 Алгоритми задач проектування

Поняття алгоритму є одним із засадничих понять сучасної науки і техніки. Алгоритм - цей опис набору правил, в результаті виконання яких від початкових даних переходять до шуканого результату. Ланцюжок дій називають алгоритмічним процесом, а кожну дію - елементарним кроком або просто кроком алгоритму.

Алгоритми класифікуються залежно від типів задач, для вирішення яких вони призначені. На основі цих алгоритмів створюються стандартні програми, що об'єднуються в ППП. Ці програми в якості початкових даних повинні отримувати відомості про ОП.

Такі дані готуються відповідно до алгоритмів обробки інформації. Вони дозволяють шукати, сортувати, перекодувати, коригувати інформацію, необхідну для розрахунків. Алгоритми проблемної орієнтації описують рішення загальних інженерно - технічних задач: статистична обробка результатів, графічне представлення результатів обчислення. Алгоритми предметної орієнтації спрямовані на отримання результатів про ті або інші об'єкти проектування. Алгоритми рішення системних задач ЕОМ створюються для організації роботи ОС, обробки файлів, управління процесом обчислення.

3 Математичне моделювання в САПР

3.1 Загальні положення

Для створення теорії математичного моделювання необхідно зрозуміти ланцюг моделювання й представити в математичному вигляді об’єкт моделювання (ОМ) [10], [21].

Слово “модель” походить від латинського “modus” (копія, образ, обрис). Приклади моделей: географічні й топографічні карти, структурні формули в хімії, книги (Знакові моделі знань), тощо. Модель - це засіб пізнання, що стоїть між логічним мисленням і досліджуваним процесом. Тобто модель - це спорудження, пристрій, сума логічних подань, що відтворює явище, подібне досліджуваному.

Моделювання - це заміщення деякого об’єкта А іншим об’єктом В. Заміщаємий об’єкт, називають оригіналом. Об’єкт,одержаний в результаті заміщення - моделлю. Тобто модель - це заступник оригіналу. Залежно від мети заміщення одного і того ж того самого оригіналу, модель може бути різною.

Основна мета моделювання в науці й техніці - вивчення оригіналу по його більш простій моделі.

Процес моделювання складається з наступних етапів:

- постановка задачі та визначення конкретних властивостей ОП і відносин між його складовими;

- визначення об’єкту моделювання;

- вибір моделі, що підлягає дослідженню.

Математична модель - це наближене, виражене в математичних термінах, представлення об’єкта проектування.

Об’єкти, що підлягають моделюванню, називаються об’єктами моделювання (ОМ).

Як правило, при моделюванні свідомо нехтують більшістю взаємозв’язків ОМ з іншими об’єктами. ОМ розглядають як окрему систему, але для корекції відірваних зв’язків вводиться принцип селективності, що забезпечує вибір необхідних зв’язків із зовнішнім середовищем. Наприклад: при моделюванні електронних схем нехтують тепловою, акустичною й механічною взаємодією, розглядаючи тільки електричні змінні. Принцип селективності вводить у систему імітацію помилок, тобто різницю в поведінці моделі та ОМ.

Принцип причинності зв’язує в системі вхідні й вихідні змінні. Для кількісної оцінки вводять поняття “стан”. Наприклад, для електронної схеми під станом розуміють значення напруг і струмів у цей момент часу.

Кінцева мета створення ММ - встановлення функціональних залежностей між змінними, котрі для кожної окремої моделі приймають строго певний вигляд. Створення ММ вимагає знання присутніх у системі елементів і взаємозв’язків між ними, процесу диференціювання й інтегрування, законів часткових областей: Кірхгофа, Ньютона, Фур’є й т.д.

Будь-яку ММ можна одержати в результаті:

1) прямого спостереження явища, прямого його вивчення й осмислення (феноменологічна модель);

2) деякого процесу дедукції, коли нова модель виходить, як окремий випадок, з якоїсь більш загальної моделі (асимптотична модель);

3) деякого процесу індукції, коли нова модель є узагальненням елементарних моделей (модель ансамблів).

Всі системи існують у просторі й часі, це значить, що час і три просторові змінні можуть розглядатися в якості незалежних змінних. Але облік цих чотирьох змінних ускладнює модель, тому найчастіше при моделюванні практикують введення лише одної або двох змінних. Приклад: будь-яка динамічна система, ММ якої є система диференціальних рівнянь із незалежною змінною t.

Існує багато ознак класифікації ММ за ознаками незалежних змінних або за ознаками представлення незалежних змінних:

1) моделі з розподіленими параметрами (всі незалежні змінні беруться в безперервній формі);

2) моделі із зосередженими параметрами (просторова змінна - дискретна, а тимчасова - безперервна);

3) моделі з дискретними параметрами (всі незалежні змінні беруться в дискретній формі).

Розглянемо більш детально класифікацію моделей взагалі та класифікацію математичних моделей зокрема.

3.2 Класифікація моделей

Загальна класифікація існуючих моделей наведена на рисунку 1.

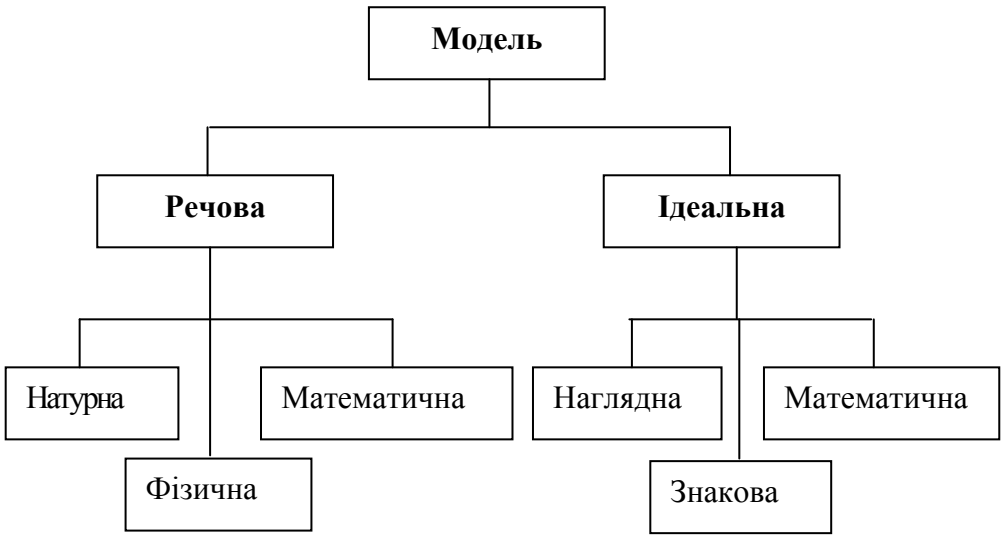
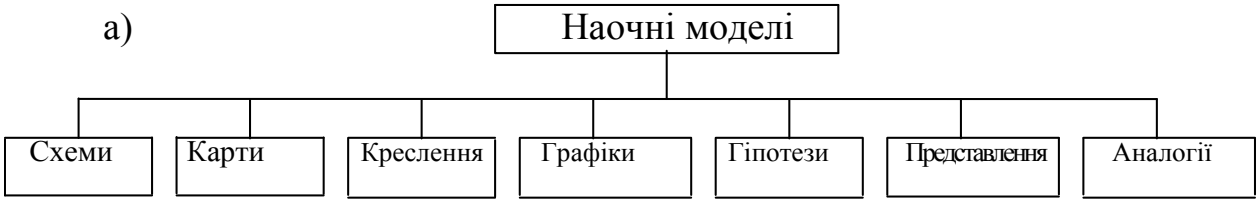


Рисунок 1 - Загальна класифікація моделей

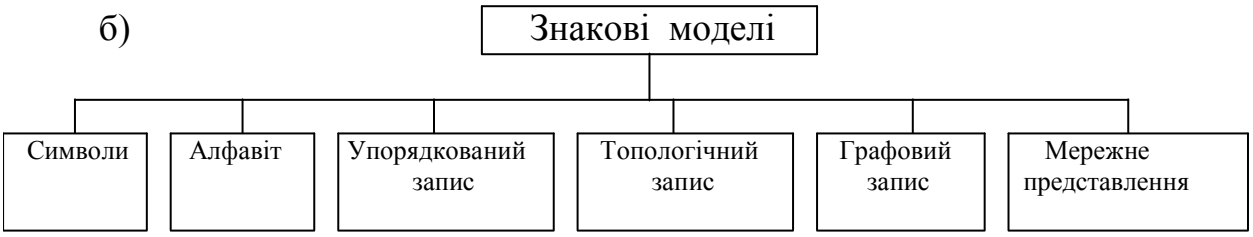
Ми будемо розглядати тільки ідеальні моделі, тому що вони відображають реальну дійсність, яка відповідає реальним умовам проектування ОП. Ідеальні моделі існують лише в пізнанні людей і функціонують за законами логіки.

У свою чергу, продовження класифікації класу ідеальних моделей представлено рисунками 2: а), б), в).

а)



б)



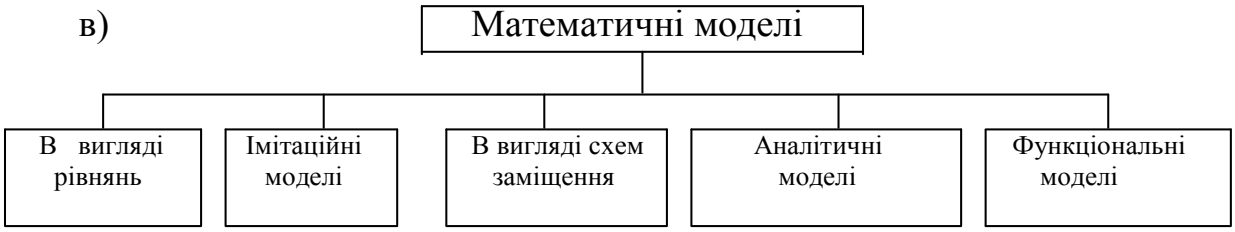


Рисунок 2 - Класифікація ідеальних моделей

Знакові моделі - це впорядкований запис символів, взаємодіючих між собою не за законами фізики, а за правилами, установленим у даній області знань. Дуже широко поширені. Практично кожна область знань - програмування, електроніка й інші - виробила свою символіку для опису моделей. Приклад: програми, схеми й т.д.

Розглянемо моделі, які мають місце в САПР ВЕТ: структурні, функціональні, геометричні, знакові, уявні, аналітичні, чисельні й імітаційні.

Структурні схеми відтворюють склад ОМ, його розташування в просторі й взаємозв’язку складових елементів, тобто - структуру системи. Можуть бути речовими (макети) і ідеальними (топологія друкованої плати).

Геометричні моделі відображають тільки структуру об’єкта й мають велике значення в проектуванні електронних схем. Вони побудовані на основі геометричної подоби й вирішують задачі оптимального розміщення ОМ (трасування друкованої плати).

Уявні моделі - це продукт чуттєвого сприйняття й тривалого абстрактного мислення. Виражені у вигляді

словесного або знакового опису. Приклад - відома планетарна модель атома Бору.

Аналітичні моделі - явна залежність необхідних величин від параметрів і змінних, що характеризують досліджуване явище.

Імітаційні моделі - реалізуються на ЕОМ у вигляді моделюючих алгоритмів (програм), що дозволяють обчислити значення вихідних параметрів. Моделюючий алгоритм не залежить від типу інформації, яку необхідно одержати в результаті моделювання.

Функціональні моделі імітують тільки спосіб поводження оригіналу, його функціональну залежність від навколишнього середовища. Найбільшу ефективність мають моделі, побудовані за концепцією "чорного ящика" (відтворюється оригінал, повністю відокремлений від його змісту й структури, всі зв'язки реалізовані математичними співвідношеннями).

3.3 Класифікації математичних моделей

ММ - універсальна, гнучка й ефективна. Представляється в абстрактній математичній формі за допомогою змінних, параметрів, рівнянь і нерівностей.

В ММ входять наступні елементи: змінні, константи (фіксовані параметри), математичні вирази, логічні вирази, інформація (алфавітно-цифрова й графічна).

Процес моделювання має логічну схему, показану на рисунку 3.

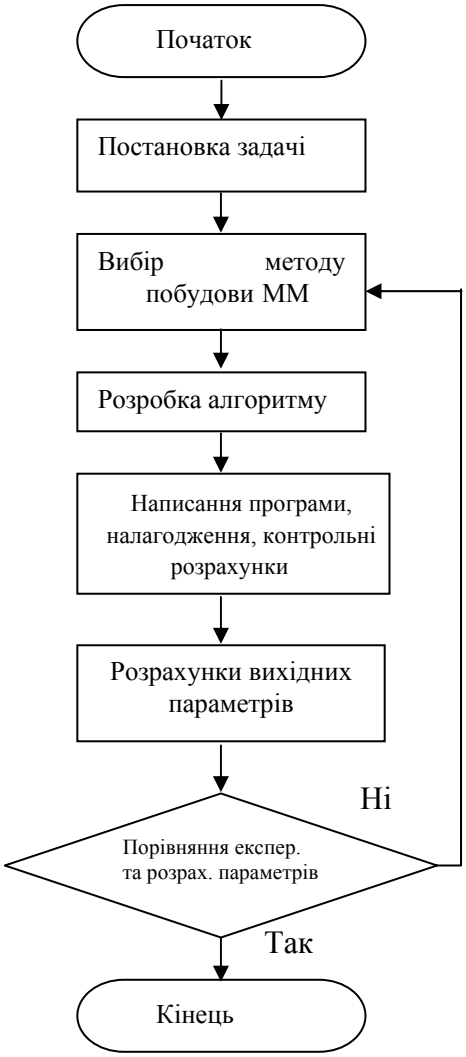


Рисунок 3 - Блок-схема алгоритму ітераційного процесу

ММ може бути також, як і ОП, обраною із числа аналогів або створена знову.

ММ класифікуються за наступними ознаками:

- за класами - речові й ідеальні;

- за типами - у вигляді рівнянь, імітаційні, у вигляді схем заміщень, аналітичні, функціональні;

- за критеріями: критерій 1 - поведінка ММ у часі; критерій 2 - по типах параметрів (елементів) ОМ; критерій 3 – по кількості та розподілу параметрів ММ та побудови її

структури; критерій 4 - за типом використовуваного математичного апарата.

Класифікація ММ за критеріями зазвичай є найбільш поширеною і використованою в САПР, оскільки надає найбільш детальну інформацію про ОП.

3.3.1 Критерій 1

Згідно класифікації по критерію 1 ММ розподіляють на наступні:

- динамічні (час відіграє роль незалежної змінної й поведінка системи в часі змінюється);

- статичні (поведінка системи від часу не залежить);

- квазістатичні (поведінка системи змінюється від одного статичного стану до іншого згідно зовнішнього впливу).

Динамічні моделі, у свою чергу, підрозділяються на:

- миттєві (поведінка в кожний момент часу залежить від зовнішніх впливів);

- з пам’яттю (поведінка в кожний момент часу визначається зовнішніми, існуючими в попередній момент часу, впливами).

Динамічні ММ із пам’яттю можуть бути:

- стаціонарними;

- нестаціонарними.

3.3.2 Критерій 2

Елементами ММ є змінні, параметри, математичні зв’язки й інформація. Проведемо класифікацію по елементах:

- детерміновані - елементи чітко встановлені, і поведінку системи можна визначити;

- стохастичні - елементи частково (або всі) не встановлені, і поведінку системи визначити не можливо;

- безперервні - якщо інформація й параметри безперервні, а математичні зв’язки стійкі;

- якщо інформація й параметри є дискретними величинами, а математичні зв’язки не стійкі, то ММ визначається як дискретна. Відповідно до цього визначення, стохастична ММ є дискретною;

- з фіксованими параметрами - якщо параметри ММ не змінюються в процесі моделювання відповідно до поведінки ОМ;

- зі змінними в часі або в просторі параметрами.

3.3.3 Критерій 3

Згідно критерію 3 ММ бувають наступними:

- з розподіленими параметрами – коли є одна або декілька незалежних просторових змінних, а інші параметри знаходяться в прямій залежності від них. ММ з розподіленими параметрами реалізуються у вигляді диференційних рівнянь;

- з зосередженими параметрами - якщо є незалежні змінні тільки в часі. Реалізуються у вигляді різниці рівнянь.

ММ також підрозділяються на:

- прості (модель одного елемента, наприклад - резистор);

- складні - елементарна підсистема, яка утримує декілька простих елементів;

- комплексні - N-і кількість елементарних підсистем.

Відповідно до реалізації структури, ММ бувають:

- деревовидні – структура ММ сформована у вигляді розгалуженого дерева, яке не має замкнутих контурів;

- мережна - якщо при формуванні структури ММ утвориться хоча б один контур.

Якщо є деревоподібна або мережна структура, то зв’язки між елементами описуються матрицями або лінійними алгебраїчними рівняннями.

3.3.4 Критерій 4

ММ може бути корисна, якщо вона веде до математичної проблеми, що дозволяє одержати чіткі вихідні результати у вигляді чисельних розрахунків. Тип розв’язуваної математичної проблеми залежить від ОМ і мети моделювання. Іноді ММ має аналітичне рішення і являє собою досить грубе наближення до проблеми, яку необхідно розв’язати .

ММ може мати лінійні й нелінійні складові й розподілятися на типи залежно від математичної проблеми:

- рівняння;

- апроксимаційні задачі;

- задачі оптимізації;

- стохастичні проблеми.

Відповідно до цього, виділяють математичні моделі наступного типу:

- оптимізаційні - що поліпшують, за рахунок спеціально введених параметрів, структуру та вихідні характеристики ММ;

- не оптимізаційні - якщо поліпшень не було.

3.4 Методи одержання ММ

Процес створення й рішення ММ є ітераційним і включає набір кроків (рисунок 3):

1) постановка задачі моделювання відповідно до наміченого ОМ, тобто - розробка ТЗ;

2) вибір методу побудови ММ;

3) розробка чисельного алгоритму рішення отриманої ММ;

4) написання реалізуючого чисельного алгоритму та його опис програмними засобами (мовою високого рівня), налагодження написаної програми, проведення контрольних розрахунків;

5) проведення розрахунків за допомогою програмних засобів для одержання вихідних параметрів;

6) порівняння експериментально отриманих і розрахункових параметрів;

7) пошук нової ММ при значній розбіжності результатів і перехід до кроку 3.

Розглянемо приклад побудови ММ у випадку реалізації фізичних систем. Основними елементами моделей фізичних систем з розподіленими або зосередженими параметрами служать фізичні явища.

Дані елементи мають відповідну орієнтацію. Для розподілених моделей орієнтацією є просторові координати, для моделей із зосередженими параметрами - наявність

вхідного й вихідного полюсів. В якості модельних елементів приймаються розсіювачі енергії, накопичувачі електричної енергії, накопичувачі магнітної енергії.

Допустимо, що перший тип елементів включає елементи із втратами. Наприклад, у випадку електричного ланцюга для розподіленої моделі елемент розсіювання в одномірному наближенні представляється зв'язком:

$$RI = -(U / x), \qquad (1)$$

де R - резистивна змінна, яка визначає ступінь розсіювання; U – визначає ступінь спадання напруги на змінній; I – значення струму через змінну; x - просторова координата, що відображає спрямованість розподіленого елемента.

Мінус показує на властивість втрати енергії.

Для систем із зосередженими параметрами залежність (1) перетворюється у відомий закон Ома:

$$U = IR, \qquad (2)$$

де R - опір резистора; I - струм через резистор; U - спадання напруги на резисторі.

Інші типи елементів представлені в таблиці 1.

Першим кроком побудови ММ є складання рівнянь, що зв'язують всі змінні і елементи, початкові й граничні умови. Основним принципом побудови даних рівнянь для електродинамічних систем повинен бути принцип збереження енергії. Для електричної системи - це перший закон Кірхгофа для струмів і другий закон Кірхгофа - для напруг. Для системи

з багатьма змінними перший і другий закон Кірхгофа мають матричний вигляд.

Таблиця 1 – Перелік складових елементів ВЕТ (ЗОТ.

Об’єкт моделювання	**Електрична система**
Розсіювач енергії	Резистор
Модель	Опір
Накопичувач електричної енергії	Конденсатор (електричний)
Модель	Ємність
Накопичувач магнітної енергії	Котушка індуктивності
Модель	Індуктивність

Другим кроком побудови ММ є визначення початкових і граничних умов. Як тільки ММ отримана, відразу виникає проблема підбору або створення математичного апарата для реалізації обчислювального процесу. Виробляється оцінка: існування рішення; стійкість обчислювального процесу й самого рішення; облік помилок; оптимальність обчислювального процесу; наявність обчислювальних засобів; можливість зміни первісної ММ для одержання розв’язку задачі.

Розглянемо докладніше розробку обчислювального алгоритму рішення створеної ММ. Обчислювальним алгоритмом називають строгу послідовність логічних і/або арифметичних операцій, що забезпечують наближення до рішення. Він може бути циклічним або ітеративним, тобто - чисельний процес може початися з якогось початкового

наближення й з кожним наступним кроком наближатися до рішення, причому - кожна ітерація складається з однієї й тієї ж послідовності кроків.

Виділимо наступні алгоритми рішення: систем алгебраїчних лінійних і трансцендентних (нелінійних) рівнянь; завдань про власні значення й власні вектори, апроксимації; інтегральних рівнянь; завдання Коша для систем звичайних диференціальних рівнянь; крайових завдань для систем звичайних диференціальних рівнянь у частинних похідних.

Побудуємо обчислювальний алгоритм (ОА). За основу приймемо ітеративний алгоритм. Такий ОА починається з деякого початкового наближення xk , де x - вектор залежних змінних, індекс k вказує номер ітерації (k = 0, 1, 2,...). Від ітерації до ітерації маємо:

$$Xk \rightarrow xk+1$$

Ітеративний процес може включати й генерувати на кожній ітерації k послідовності z_0, z_1,…z_N із умовою вибору найкращого наближення x_{k+1} з множини $\{Z\}$. На кожній ітерації повинна виконуватися спеціальна перевірка на закінчення ітеративного процесу або у випадку одержання дійсного рішення x, або у випадку неможливості одержання точного рішення.

Будь-якому розробнику ОА необхідно вирішити наступні важливі проблеми:

- чи сходиться дана послідовність x_0, x_1,...x_k до рішення;
- якщо сходиться, то як швидко й до якого рішення;
- яка при цьому допущена помилка.

Відповіді на ці питання забезпечують рішення поставленої задачі. Ефективність алгоритму безпосередньо пов’язана з матеріальними витратами на розробку математичного моделювання, тому що вартість машинного часу залежить від витрат на одну ітерацію й від числа ітерацій. Будь-який ОА можна оцінити в елементарних операціях (додаваннях і множеннях) і в такий спосіб зв’язати з обчислювальними ресурсами. Мета й обов’язок будь-якого розроблювача ММ і ОА - мінімізувати обчислювальні ресурси.

КОМПОНУВАННЯ Й РОЗМІЩЕННЯ ЕЛЕМЕНТІВ У МОНТАЖНОМУ ПРОСТОРІ

1 Загальні положення

Загальна задача синтезу топології визначається, як сукупність наступних задач: компонування, розміщення, трасування [21].

При заданій електричній схемі й бібліотеці елементів, метричних і топологічних обмежень, у першу чергу потрібно знайти найкраще розміщення елементів і з'єднань, тобто – виконати задачу компонування.

1.1 Задача компонування

Бібліотека елементів містить типові зображення елементів у вигляді блоків (макросів), що враховують, як габаритні розміри, так і установочні розміри ЕРЕ. Наприклад, на рисунку 1а показаний резистор типу МЛТ-0,125, відповідно блок (макрос) цього елемента буде мати вигляд, зазначений на рисунку 1б [21], [12].

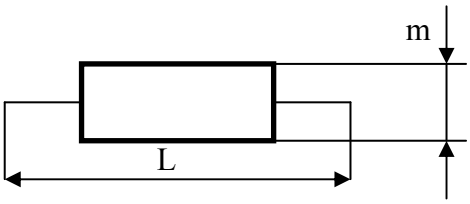


Рисунок 1а

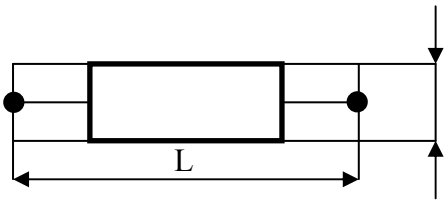


Рисунок 1б

Блок може поєднувати й кілька елементів (тригер, суматор і т.д.). Основна задача компонування - розподілити елементи електричної схеми так, щоб число зв'язків між блоками було мінімальним.

Окрім критерію зв'язаності, часто потрібно провести вибір такого компонування деяких блоків, яке дозволило б реалізувати завершену логічну функцію (підсилювач, стабілізатор і т.д.).

Обмеженням у даній задачі є ємність блоків, тобто, число призначених в них елементів і виводів блоку з урахуванням індивідуальних характеристик, наприклад, їхньої площі. Задачу

компоновки називають також задачею декомпозиції, тому що багаторазове її рішення дозволяє одержати ієрархічну структуру об'єктів, або ж задачею попереднього розміщення. Алгоритми компонування аналогічні алгоритмам розміщення.

1.2 Задача розміщення

Задача розміщення - оптимально розмістити блоки в заданій області монтажного простору по сукупності критеріїв якості із задоволенням конструкторсько-технологічних вимог.

Найбільш загальне обмеження - відсутність перекриття меж елементів, причому між елементами повинна залишатися відстань не менша визначеної технологічної константи. Крім того, не слід поруч розташовувати елементи визначеного типу, наприклад, потужні активні елементи для забезпечення рівномірності теплового поля та напівпровідникові елементи, тощо.

Найважливішим критерієм оптимальності розміщення є створення умов для наступного трасування. Оскільки трасування виконується після розміщення, його критерії носять орієнтовний характер. Важливо так розмістити елементи, щоб на етапі трасування вдалося досягти максимальної щільності упаковки.

1.3 Щільність упаковки

Щільність упаковки прораховується по формулі

$$f = (S_e + S_c) / S_к,$$

де S_e, S_c, $S_к$ - площі, зайняті елементами, з'єднаннями, кристалами. Щільність упаковки контролюється на всіх етапах розміщення, як параметр.

З огляду на обмеження, плата вважається оптимальною, якщо щільність упаковки становить не менш 84%.

2 Розміщення одногабаритних елементів

При рішенні завдання розміщення одногабаритних елементів звичайно використається регулярний монтажний простір і модель ОП у вигляді графа.

Завдання полягає в розташуванні вершин графа в позиції простору так, щоб можна було мінімізувати сумарну довжину ребер.

Основними етапами рішення задачі розміщення є: вибір критеріїв розміщення, початкове розміщення елементів на ДМП, ітераційна оптимізація початкового розміщення. Однак через умовність розподілення задач розміщення й трасування побудувати подібний критерій, що досить точно відображає умови прокладки з'єднань на ДМП, важко. Але можливо й необхідно.

Задача розміщення одногабаритних елементів на друкованій платі в 96% випадків є окремим випадком задачі розміщення різногабаритних елементів. Тому методи її реалізації розглянемо коротко.

Розглянемо послідовний алгоритм, основною ідеєю якого є ідея впорядкування вершин за певними ознаками. У встановленій черговості для кожної з них визначається

найкраща позиція (наприклад; по сумарній довжині ребер із уже розміщеними вершинами). Потім процес повторюється для вершин, що залишилися, і вільних позицій.

Для поліпшення рішення часто використовують ітераційні алгоритми, побудовані на основі парної або групової перестановки вершин. Дві вершини переставляються, якщо це зменшить сумарну довжину ребер графа. Існуючі алгоритми, в основному, відрізняються способами вибору пар для пробного обміну.

3. Задачі розміщення різногабаритних елементів

Найбільш ефективно дана задача вирішується при використанні безперервного монтажного простору. У загальному випадку задача полягає у визначенні координат макросів таких розмірів, щоб: кожен макрос, що відображає елемент, був розташований у прямокутній області розміщення A; будь-які два прямокутники не перетиналися та знаходилися один від одного та від межі A на відстані d; орієнтовані компоненти були встановлені в заданій орієнтації; зазори між сусідніми прямокутниками незначно відрізнялися від заданої величини d0; щільність заповнення A була близькою до постійної величини; щільність A була близькою до максимальної; розміри всіх прямокутників (макросів) лежали в заданих межах і мінімально відрізнялися від вихідних; забезпечувалися найбільш сприятливі умови для наступного етапу трасування з’єднань між компонентами.

Рішення цієї задачі виконується у два етапи. На першому етапі визначається відносне розташування елементів по їхніх ідеальних координатах (етап попереднього розміщення). На другому етапі розміщаються макроси й вибирається їхня форма з урахуванням ідеальних координат. Як правило, при перевірці можливості розміщення елемента враховується можливість розміщення зазорів до d, можлива деформація макросу і його поворот. Перекриття елементів виключається. Якщо припустиме рішення не знайдене, необхідно провести згладження межі за рахунок зменшення вільної області ДМП.

4. Компонування блоків

Як модель об'єкта використовується звичайно гіперграф. Задача полягає в розрізуванні ОП на частини таким чином, щоб мінімізувати число з'єднань між отриманими в результаті декомпозицій блоками при певних обмеженнях.

У послідовних алгоритмах компонування блоків виконується їх послідовним заповненням (алгоритм заснований на використанні методів послідовних наближень).

4.1 Метод послідовних наближень

Суть методу послідовних наближень укладена в наступному:

Нехай q_0 - початкове розміщення елементів ОП; $Q_i(q_i\text{-}1) = q_i$ - розміщення на і-м кроці оптимізації; $F(q_i)$ - значення цільової функції.

Тоді завдання ітераційної оптимізації розміщення зводиться до вибору послідовності перетворень $q_0, Q_1(q_0)=q_1$, $Q_2(q_0)=q_2$, і т.д., для якої

$$F > F(q_1) > F(q_2) > ... > F(qs).$$

В якості цільової функції F можна взяти один із критеріїв розміщення, розглянутих нами раніше.

5 Дискретні методи рішення задачі синтезу

Задача компонування й розміщення ЕРЕ можуть бути представлені як об'єкт, до якого застосовні загальні методи дискретної математики. Методи перетворення одних дискретних множин в інші називають дискретними. Розглянемо найбільш поширенні дискретні методи рішення задач синтезу: метод вектора спаду, метод гілок та меж, метод послідовного аналізу, мето Парето.

5.1 Метод вектора спаду

Загальну задачу дискретної оптимізації можна поставити в такий спосіб. Задано:

- Q - дискретна множина;

- ρ - його метрика;

- $F(x)$, x це Q - дійсна функція.

Потрібно на деякій підмножині $Q1 \leq Q$ знайти точку x_0 мінімуму функції $F(x)$:

$$F(x_0)=min\ F(x)\ * \qquad (1)$$

Задачу максимізації можна звести до (1), замінивши $F(x)$ на - $F(x)$. Якщо $Q_1<Q$, - метод локально оптимальний, якщо $Q_1=Q$, - метод точний.

Тобто, реалізується стратегія часткового перебору, результатом якої буде побудова послідовності точок локального оптимуму x_1, x_2 на околицях $Q(k)$. Рішення закінчується, якщо:

а) на деякому кроці $xi=x_{i-1}$;

б) $F(x_i\text{-}1) - F(x_i) / F(x_i) \leq \delta$, де δ – задана відносна точність рішення;

в) $i = n$, де n-задане число ітерацій.

Вище описаний метод одержав назву методу вектора спаду. Застосовується до рішення задач компонування й розміщення.

Метод вектора спаду дає грубе рішення. Підвищення точності можливе при багаторазовій реалізації цього методу при різних початкових умовах, а це вимагає значних витрат машинного часу.

5.2 Метод гілок і меж

Метод гілок і меж - один з найпоширеніших точних методів дискретної оптимізації, застосовуваних при рішенні завдань компонування, розміщення й трасування.

Суть методу розглянута нами в лекції «Синтез і аналіз». Особливість його застосування полягає у виборі оцінок підмножин для кожної конкретної задачі, звичайно при зміні

задачі змінюється лише цільова функція й критерії (метричні й топологічні обмеження).

Метод гілок і меж доцільно застосовувати при числі елементів, що не перевищує 20 одиниць, тому що отримані розрахунки громіздкі, обчислення тривалі й дорогі через великі витрати машинного часу.

Вищеозначені недоліки присутні і методу гілок та меж, методу Парето.

6 Спеціальні алгоритми

В САПР частіше застосовуються спеціальні алгоритми, тому що вони дають кращі результати при великих значеннях n (кількість елементів). Ці алгоритми діляться на два класи: конструктивні й ітераційні [21], [12].

Конструктивні алгоритми використовуються для формування компонування й розміщення ЕРЕ на ДМП на основі заданих множин елементів і блоків, матриць зв'язності і відстаней. При цьому частина елементів може мати або не мати призначення в блоки. Ітераційні алгоритми формують розміщення на основі початкового розміщення (заданого раніше) шляхом послідовних локальних його поліпшень. Вони призначені для поліпшення компонування (розміщення), отриманої в результаті роботи конструктивного алгоритму.

В САПР задачі компонування й розміщення вирішують шляхом використання як конструктивних (на першій стадії), так і ітераційних (на другій стадії) алгоритмів з можливістю часткового призначення елементів у блоки.

6.1 Конструктивні алгоритми

Найбільш яскравими представниками цього класу є: алгоритм декомпозиції Кодреса й типовий конструктивний алгоритм розміщення.

Алгоритм декомпозиції Кодреса

У розглянутій постановці задачі декомпозиції враховуються не тільки зв'язаність, але й індивідуальні характеристики елементів і множин, площа й число зовнішніх виводів. Нехай вихідні дані мають такий вигляд:

- *S* - площа, відведена під розміщення;

- *E* - число зовнішніх ланцюгів;

- *X*1 - множина елементів (l_n);

- *X*2 - множина ланцюгів;

- *Y* - множина ребер;

- *A* - область розміщення внутрішніх ланцюгів, *B*(*A*) зовнішніх;

- *i* - множина внутрішніх ланцюгів

Площу й число зовнішніх виводів визначаємо за формулою:

$$S(A_i) = \sum_{X=A_i} S(x) \qquad (2)$$

$$E(A_i) = \sum_{X=A_i} E(x) - \sum d(t)$$

Нехай S,E - верхні припустимі значення, площі й числа зовнішніх зв'язків будь-якого блоку.

Розбиття

$$x1 = \underset{i=1}{U} A_i, \quad A_i \cap A_j = \varnothing \ \ (i \neq j) \qquad (3)$$

називають припустимим, якщо виконується умова

$$S(A_i) \le S, \ E(A_i) \le E \ \text{ для всіх i=1,... k.}$$

6.2 Ітераційні алгоритми

6.2.1 Силовий алгоритм релаксації

У цьому алгоритмі для кожного елемента М обчислюється силовий вектор

$$\overline{F}_{м} = \sum_i C\mu_i \overline{S\mu_i} \qquad (4)$$

де $C\mu_i$ – вага ребра, що з'єднує елементи μ та i ; $S\mu_i$ - вектор відстані між μ_i та i.

Точка, у якій $F\mu=0$, називається точкою цілі елемента, позиція цілі - це позиція, найменш віддалена від точки цілі.

При переміщенні елемента μ у точку цілі на нього перестають діяти сили - відбувається релаксація. Якщо позиція цілі була зайнята іншим елементом, то для звільнення з позиції елемента - підраховується позиція цілі й т.д. Описані переноси утворить ланцюжок переносів, який закінчується, коли черговий елемент переноситься в незайняту позицію. Ланцюжок приймається, коли це призводить до скорочення сумарної довжини з'єднань. Ітерація закінчується, коли всі елементи проаналізовані в ланцюжках перенесень. Елементи початку різних ланцюжків звичайно вибираються за принципом зменшення інцидентних їм ланцюгів.

6.2.2 Силовий алгоритм попарної релаксації

Алгоритм аналогічний попередньому, але проба замін виконується тільки для елементів, у яких позиція цілі елемента А знаходиться в Е(В) - околиці елемента В і навпаки, позиція цілі елемента В знаходиться в Е(А) околиці елемента А. Як і попередньому алгоритмі, заміна реалізується лише в тому випадку, коли скорочується довжина з'єднань.

Ітераційні алгоритми розміщення вимагають початкового розміщення елементів. Це розміщення бажано проводити, використовуючи конструктивні алгоритми.

Ілюстрація роботи алгоритму попарної релаксації показана на графічній моделі комутаційних зв'язків ОП рисунком 2.

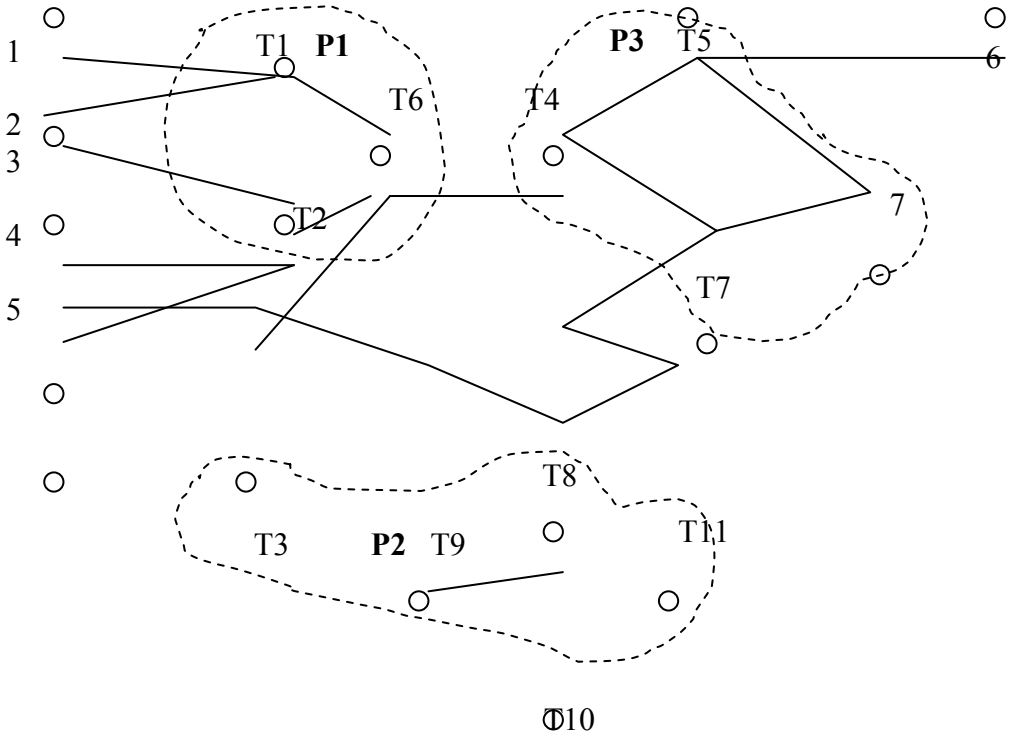


Рисунок 2 - Модель комутаційних зв'язків ОП

Робота алгоритму базується на розбитті множини елементів на мінімальне число множин (або класів) малозв'язних між собою (Р1,Р2,Р3). Обчислення провадяться для кожного класу окремо й тільки після того, як всі елементи призначені в класи, алгоритм припиняє роботу.

Умов зв'язаності елемента й класів Р1 і Р2 чотири:

- елемент непов'язаний із класами Р1 і Р2;
- елемент пов'язаний із класами Р1 і Р2;
- елемент зв'язаний лише із класом Р1;
- елемент зв'язаний лише із класом Р2.

Суть пропонованого методу полягає в наступному: для підмножини A_i <= x_i площа й число зовнішніх виводів визначаються по формулах

$$S(A_i) = \sum_{x \in Ai} S(x) \tag{5}$$

$$E(A_i) = \sum_{x \in Ai} E(x) - \sum_{t \in I(Ai)} d(t) - \sum_{t \in B(Ai)} d((t) - 1) \tag{6}$$

де $I(A_i)$ - множина внутрішніх ланцюгів, тобто підмножина елементів X_2, інцидентних (приналежних) тільки елементам підмножини A_i; $B(A_i)$ - множина зовнішніх ланцюгів, тобто підмножина елементів X_2, інцидентних хоча б одному елементу безлічі A_i, які одночасно є вхідними сигналами елемента, що не входить у цю множину; $d(t)$ – число вершин, що належать A_i й інцидентних вершині t.

6.2.3 Модифікований алгоритм сусідніх парних замін

Цей алгоритм звичайно використовується на остаточній стадії розміщення. Для нього характерні проби замін пар компонентів (елементів), що породжують паразитні елементи. Якщо пробна заміна призводить до скорочення цільової функції, то вона може бути реалізована, у противному випадку - заміна вважається неприпустимою, елемент, що породжує паразитний елемент, заміняється із сусіднім. Функція сусідства первинного елемента задається користувачем. При цьому змінюються місцями тільки елементи, які можуть бути вписані у площини, що звільняються.

6.2.4 Алгоритм ітераційного розміщення одногабаритних елементів ГОТО

Алгоритм Гото найбільш ефективний з використаного в САПР класу ітераційних алгоритмів.

Його суть складається в пошуку циклічної перестановки елементів на деякій групі позицій, що поліпшує значення цільової функції. Якщо в результаті перестановки значення цільової функції зменшується - то таку перестановку називають ефективною.

Складність алгоритму оцінюється величиною

$$N=C\varepsilon\mu\,, \qquad (7)$$

де C- середнє число операцій по визначенню ефективності однієї перестановки; ε- множина найкращих позицій цього елемента на ДМП; μ - медіана множини.

Дослідження показали,що алгоритм найбільш ефективний при ε=4, μ=4.

Обчислення закінчується, як тільки виявляється ефективність перестановки або довга перестановки стає більше припустимою. У другому випадку, за допомогою датчика випадкових чисел вибираємо й обчислюємо нову послідовність.

ЗАДАЧІ ТРАСУВАННЯ З'ЄДНАНЬ

1. Загальні положення

1.1 Ручна розробка топології

Інтерактивний режим розробки топології широко поширений через ряд переваг і, насамперед, характеризується можливістю використання евристичних прийомів. Він

забезпечує високу щільність укладання ЕРЕ на ДМП, близьку до оптимальної. Недолік - необхідна висока кваліфікація проектувальника й більша витрата часу на виконання проектних процедур. Індивідуальний стиль конструктора також накладає відбиток на проект, а це може затрудняти спряження з іншими підсистемами САПР. Крім того, ручне проектування може порушити єдину методологію проходження проектних процедур. При ручній розробці застосовуються перший і другий маршрут проектування виробів РЕА, розглянуті в попередніх темах.

1.2 Автоматизовані методи проектування топології

Загальна ідея ряду автоматизованих методів проектування полягає в описі топологічних структур спеціальними символами.

Проектувальник готовить символьну структуру, тобто - вказує положення елементів, блоків (макросів), меж з'єднань і т.д., після чого автоматично розробляється та надається в графічній формі топологічна схема ОП.

Найбільш відомі два підходи в розробці топології:

- по координатній сітці з фіксованим кроком;

- розробка "паличковими,, методами.

У першому підході вибирається фіксований крок по координатній сітці, обумовлений метричними й топологічними обмеженнями, з наступним заповненням символами комірок ДМП. Застосування цього підходу дозволяє: використати

спрощені проектні норми й виконувати контроль правильності електричних з'єднань і готових ескізів топології.

Другий підхід заснований на введенні топологічних елементів у вільній формі у вигляді ескізів. Схема розміщення елементів проводиться по координатній сітці. Окремі топологічні елементи прив'язуються до координатних ліній. В якості вхідної інформації фігурує довільно скомпонований ескіз. На виході формується топологічна документація.

Пакет програм STIK фірми SALMA виключає координатну сітку, як засіб проектування. Після введення ескізу топології в символьній формі за допомогою редактора проводиться стиснення ескізу топології з урахуванням конструкторсько-топологічних норм та обмежень.

Специфікою символьних методів є пакетний режим обробки й редагування топології, що в результаті значно скорочує трудомісткість роботи проектувальників. У порівнянні з ручними методами, час розробки топології скорочується від 50% до 70%, але щільність укладання зменшується до 20%.

1.3 Автоматичні методи проектування топології

Дані методи поки розроблені тільки для проектування елементів: ВІС, ЗВІС, МАВІС тощо. По якості проектування вони значно перевищують інтерактивні та автоматизовані методи, оскільки мають наступні переваги:

- одержання працездатності елементів у гранично стислий термін при припустимій неоптимальності площ і електричних характеристик;

- наявність апарата верифікації, що дозволяє швидко оцінити результати моделювання;

- наявність інтерактивного режиму, що дозволяє відредагувати отриманий варіант топології,в разі виявлення суттєвих недоліків, в інтерактивному режимі безпосередньо користувачем в процесі промислової експлуатації САПР.

2. Задачі й методи трасування з'єднань

Задача трасування з'єднань - побудувати в заданій області з'єднання виводів блоків (макросів), у відповідності зі схемою електричною принциповою, синтезованого оптимально по сукупності критеріїв якості та із задоволенням конструкторсько - технологічних обмежень.

Слід зазначити, що поділ синтезу топології на задачі компонування, розміщення й трасування умовний, тому що вони вирішуються в комплексі та із урахуванням загальних вимог до ОП.

Представлення про складність рішення задач трасування можна скласти, якщо розглянути найбільш простий клас з'єднань.

Нехай :# m_i - число виводів 1-го ланцюга i=1, ..., m.

Тоді число конфігурацій ланцюгів, виконаних у вигляді дерев без додаткових вершин, можна представити наступним виразом $N = \prod_{i=1}^{M} m_i^{m_i-2}$.

Цей вираз дозволяє оцінити знизу складність трасування. Більше точне значення можна одержати, помноживши N на число варіантів призначення гілок дерев в область трасування.

2.1 Хвильовий алгоритм ЛІ

Для виконання завдання автоматизованого та автоматичного трасування з'єднань використають наступні алгоритми або їхні комбінації й модифікації:

- хвильові алгоритми або пошук шляху в лабіринті;
- променевий алгоритм;
- алгоритми канального трасування.

Найбільш універсальним є хвильовий алгоритм та його модифікації. Він дозволяє знайти шлях у всіх випадках, коли шлях існує. Однак хвильовий алгоритм вимагає більших витрат пам'яті ЕОМ і машинного часу в порівнянні з іншими алгоритмами. Якість алгоритму оцінюється насамперед відсотком невідтрасованих з'єднань.

Хвильовий алгоритм ЛІ використовує дискретний монтажний простір у вигляді технологічної сітки, яку називають координатною сіткою.

ДМП складається з підмножини комірок: зайнятих і вільних. Причому, після проведення траси всі комірки, їй відповідні, будуть вважатися зайнятими.

Приклад проведення траси від точки а до точки в показаний на рисунку 1.

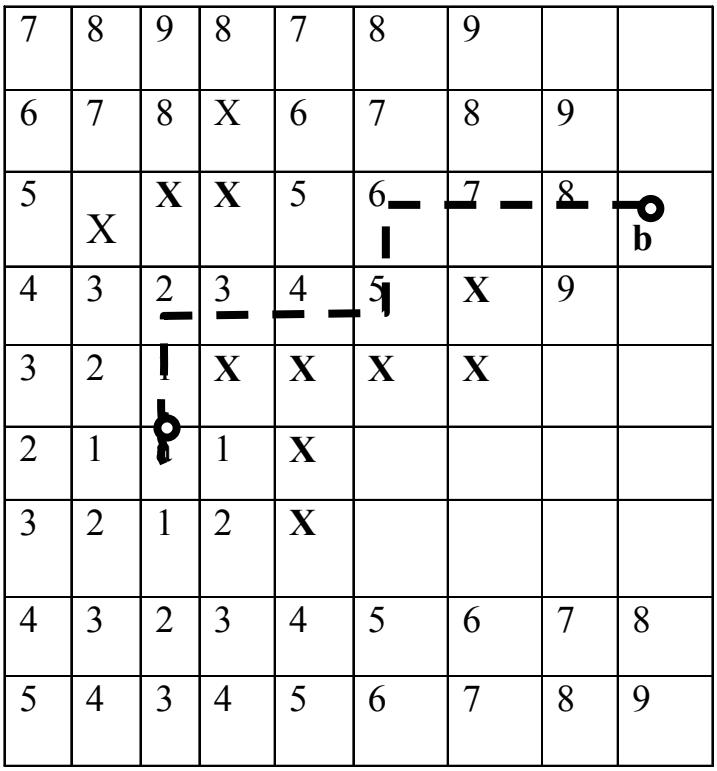


Рисунок 1- Приклад проведення траси від точки а до точки в

Припустимо, що необхідно визначити шлях мінімальної довжини між контактами а й в. Пошук шляху складається із двох етапів - поширення хвилі і моделювання шляху.

На першому етапі моделюється поширення хвилі по вільних комірках від комірки, яка утримує точку а. Точка а буде джерелом хвилі, а точка в - ціллю, тобто - хвиля буде розповсюджуватися зліва направо. Спочатку, в число зайнятих комірок (на рисунку 1 вони позначені символом ”x") розміщуємо області, заборонені для трасування: виводи елементів, власне елементи, технологічні зони й межі

комутаційних полів. У процесі роботи алгоритму до них додаються області, зайняті побудованими з'єднаннями.

У процесі поширення хвилі, деяким коміркам відповідає значення функції f(c), яка означає масу комірки. Комірка вважається не відміченою, якщо її маса дорівнює нулю. На початку шляху хвилі комірці а привласнюється маса f(a)=0. Наступним коміркам привласнюється одинична маса. Сукупність комірок рівної маси утворить фронт хвилі. Номер фронту збігається з масою комірок, які утворюють його, а хвиля представляється послідовністю фронтів Фк (до=1,2,3,…n)...

На першому кроці проглядаються й запам'ятовуються всі вільні невідмічені комірки, їм привласнюється маса f1(a)=1; на другому кроці проглядаються всі вільні комірки, сусідні з f1(a)=1, їм привласнюється маса f2(a)=2 і т.д. Поширення хвилі йде по осі х і лише у випадку зустрічної перешкоди у фронт включаються значення по осі у. Хвиля поширюється доти, поки вона не досягне комірки в або ж жодна із точок не буде включена у фронт. Це означає, що доступ до цілі відсутній і з'єднання побудувати не можна.

В цьому випадку пара точок (джерело - ціль) заноситься в список невідтрасованих з'єднань; із ДМП стирається інформація про хвилю (номери фронтів) шляхом обнуління номерів комірок і здійснюється перехід до поширення хвилі для наступної пари точок (джерело - ціль).

Якщо ж точка - ціль досягнута, то здійснюється перехід до наступного етапу - проведення шляху.

Для проведення шляху проглядаються всі відзначені комірки із комірки в і вибирається комірка з найменшою масою в1, на наступному кроці процедура повторюється для клітки в1 і т.д. доти поки не досягнемо а. При цьому перевага надається горизонтальному руху.

Суттєвим недоліком алгоритму ЛІ є:

- неможливість здійснення деформації раніше побудованих провідників(переміщення, розсування, зламу й т.д.);

- недотрасування плати (віддалених областей) складає до 10%;

- потреба в значних ресурсах пам'яті ЕОМ і машинного часу;

- облік конструкторсько - технологічних критеріїв недостатньо високий, а облік багатокритериальносьті відсутній.

Переваги алгоритму ЛІ:

- якщо між точками а і в шлях існує, то його завжди можна знайти;

- з безлічі можливих шляхів завжди буде знайдений найкоротший;

- реалізація алгоритму проста й зрозуміла;

- специфіка розташування елементів і особливості конструкції враховуються вибором відношення зайнятості, не вимагаючи модифікації алгоритму.

2.2 Модифікації хвильового алгоритму Лі

При модифікації хвильового алгоритму Лі переслідуються наступні цілі:

- зменшення необхідних ресурсів пам’яті й машинного часу;

- поліпшення обліку конструкторсько-технологічних обмежень і критеріїв;

- забезпечення обліку багатокритеріальності.

Розглянемо найбільш застосовані в діючих системах САПР РЕА (ВЕТ) алгоритми, які є модифікацією хвильового алгоритму ЛІ.

2.2.1 Метод зустрічних хвиль

По черзі моделюється кожний фронт хвилі для пари точок а й в (на кожному непарному кроці а - джерело, в - ціль; на кожному парному – навпаки: в – джерело, а - ціль). В алгоритм цього методу уведений блок перевірки умови зіткнення фронтів від різних джерел. Якщо таке є - поширення хвилі припиняється, якщо ні, алгоритм поширюється далі.

Метод зустрічних хвиль дозволяє: зменшити час поширення хвилі, тому що площа, захоплювана нею, на ДМП без перешкод зменшується вдвічі. Дійсно, якщо відстань від а до в дорівнює R, то в моделі з односпрямованою хвилею площа захоплюваного ДМП дорівнює ПR2, а в моделі із зустрічними хвилями 2П(R/2) 2=ПR2/2.

2.2.2 Обмеження області поширення хвилі

Обмеження області поширення хвилі здійснюється наступним чином. Вибирають прямокутник, що містить точки а

й в, за межами якого хвиля не поширюється. У цьому випадку зменшується час роботи алгоритму, але збільшується відсоток невідтрасованих з’єднань.

2.3 Променеві алгоритми

Цей клас алгоритмів звужує поняття хвилі до променя. Промінь визначається, як деякий список шляхових координат, що вказує пріоритет напрямків.

По числу заданих променів, алгоритми діляться на двопроменеві, трипроменеві й т.д. Основними кроками при використанні променевого алгоритму є:

- визначення променів для джерела й цілі;
- почергове поширення всіх променів на один крок спочатку від джерела, а потім від цілі з покроковою перевіркою умови на закінчення алгоритму;
- повернення променя в висхідну точку при його блокуванні;
- побудова шляху при перетинанні різнойменних променів або встановлення неможливості проведення шляху.

2.3.1 Двопроменевий алгоритм

Нехай промінь А визначає наступний порядок присвоєння шляхових координат: $\downarrow$, $\rightarrow$, тобто до точки А промінь приходить зверху, а при зустрічі з перешкодою вигин лінії відбувається вліво.

Продемонструємо роботу алгоритму прикладом, показаним на рисунках 2, 3.

B	←	←	←	←	←	X					
↑					↑	X					
↑					↑	X	X	X	X		
↑		X	X	X	X	X					
↑											
↑	←	←							X	X	X
X	X	X	X					→	→	→	↓
							→	↓			↓
							X	↓			↓
							X	↓			↓
							X	↓			↓
							X	↓	→	→	A

Рисунок 2 - Стан ДМП після N-го кроку. Промінь B1 заблокований

Визначимо для джерела A и цілі B наступні пари променів:

A1 :↓,→; A2 :→, ↓

B1 :↑,←; B2 :←,↑

Таке визначення викликане географічним розташуванням точки A (нижче й правіше точки B).

На рисунку 2 показаний стан ДМП після N-го кроку. Промінь B_1 заблокований, а на рисунку 3 – стан ДМП після закінчення роботи алгоритму. Після (N+4) кроку в комірці P (координати 5,5) зустрічаються промені A2 і B1. Із цієї точки по шляхових координатах будують відрізки шляху від A до B.

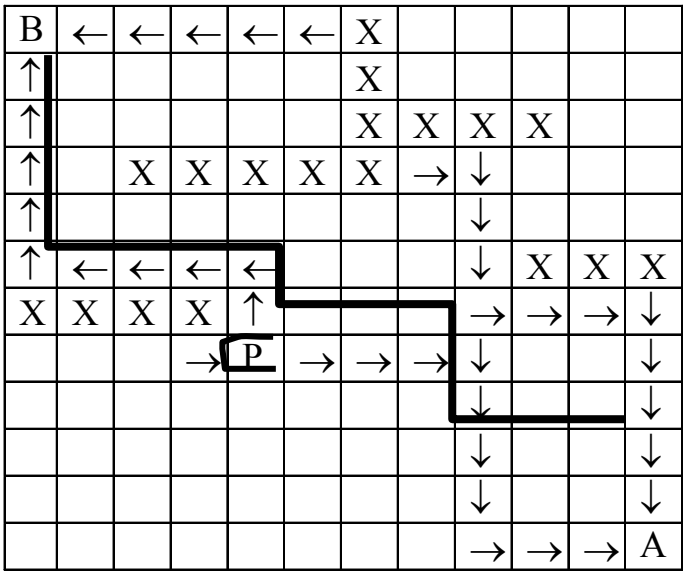


Рисунок 3 - Стан ДМП після закінчення роботи алгоритму

Якщо промені виявилися заблокованими, то здійснюють процедуру виводу променя з тупика. Із заблокованої комірки промінь повертають назад у попередню. При цьому блоковану комірку вважають зайнятою для подальшого поширення. Далі, використовуючи шляхові координати променя з меншим пріоритетом, виводять промінь із тупика. Якщо це не вдається, то промінь продовжує повертатися назад і число блокованих комірок росте. Для виходу променя з тупика за n кроків повернення, потрібно зробити n+1 додаткових кроків розповсюдження променя для вирівнювання довжин променів. Якщо промінь неможливо повернути в точку А, то проведення шляху неможливе.

Променеві алгоритми мають певні переваги й недоліки в порівнянні із хвильовими, а саме:

- хвильовий алгоритм на відміну від променевого без обмеження області поширення хвилі гарантує побудову шляху, якщо він існує;

- час роботи променевого алгоритму менший, ніж хвильового алгоритму;

- променевий алгоритм працює повільніше на завантаженому ДМП і швидше на вільному, а хвильовий - навпаки.

Тому, на першій фазі трасування рекомендують використовувати променеві алгоритми, а на другій - хвильові. Але стадію повернення променевого алгоритму з тупика варто обмежувати, тому що в складному лабіринті час роботи цього алгоритму різко зростає.

2.4 Канальне трасування

Алгоритми канального трасування застосовують у тих випадках, коли точне розташування блоків (макросів) невідоме, а траси мають переважний напрямок (матричні ВІС на основі стандартних блоків).

Канальні алгоритми базуються на основі представлень про магістралі й канали.

Магістраль - відрізок прямої, по якій може проходити з'єднання в переважному напрямку. Канал - область на кристалі прямокутної форми, на одній або декількох сторонах якої розташовані контакти (виводи з'єднань) із системою однаково спрямованих магістралей.

По напрямку магістралей розрізняють горизонтальні й вертикальні канали, а по числу сторін, на яких є канали - одно - ,2 - х - , 3 - х - , 4 - х - сторонні канали.

Основне завдання канального трасування - вибір найменшої ширини каналу, достатньої для розміщення в ньому всіх з’єднань і призначення з’єднань на магістралі.

При двошаровому трасуванні, горизонтальні ділянки ланцюгів розташовуються в одному шарі, а вертикальні - в іншому.

Алгоритми канального трасування є вузько - спеціалізованими, тому більш докладний їхній розгляд при вивченні дисципліни ‘‘ОАП ЗОТ’’ не передбачено.

2.4 Методика проектування на основі стиснення рисунку топології

В основу даного напрямку при автоматизованому проектуванні покладена імітація ручного проектування топології. Передбачається пересування всіх основних елементів топології (трас, компонентів, міжшарових переходів) і зміна конфігурації трас під час стиснення.

Спочатку виконується ескізне трасування в укрупненому дискретному полі (ДРП). При цьому вважається, що розміри одної укрупненої дискрети дорівнює розмірам максимального міжшарового переходу або траси максимальної ширини.

Допускається: однакова висота всіх компонентів, розташування нагорі (унизу) всіх виводів, однакова ширина всіх трас, застосування міжшарових переходів у місцях зміни напрямку трас, збіг розмірів міжшарового переходу із шириною трас, проведення між комірками однієї лінійки необхідного числа трас.

При даних допущеннях, завдання синтезу топологія спрощується. Етапи синтезу топології (розміщення й трасування) були розглянуті раніше. Очевидно, що для них можна застосовувати найбільш прості й швидкі алгоритми. Потім отриманий ескіз піддається стисненню з усіх боків до геометричного центра ескізу топології. При цьому комірки позбуваються своїх меж і стиснення виконується на рівні компонентів. Допускається 4 напрямки стиснення: вниз, нагору, вліво, вправо. При стисненні в кожному напрямку застосовують:

- зсув компонентів і з'єднань;
- скорочення петель з'єднань.

У результаті з'являється ескіз топології, що враховує технічні вимоги, але відображений в укрупненому ДМП. Здійснюємо перехід від укрупненого ДМП до реального. Ширина дискрети визначається мінімальною шириною траси або мінімально припустимою відстанню між компонентами. Для ДМП використовують моделі компонентів топології, серед яких з'являється новий компонент - міжшаровий перехід. Вводяться також технологічні й конструкторські вимоги. Це дозволяє стиснути малюнок топології з урахуванням технічних вимог.

CASE-ТЕХНОЛОГІЯ ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ІНФОРМАЦІЙНИХ СИСТЕМ (ІС)

1. Вступ

САПР, згідно класифікації, відносяться до класу інформаційних систем (ІС). Інформаційна система (Information system) — сукупність організаційних і технічних засобів для збереження та обробки інформації з метою забезпечення інформаційних потреб користувачів. Таке визначення може бути задовільним тільки при найбільш узагальненій і неформальній точці зору і підлягає подальшому уточненню. Інформаційні системи діють в Україні під назвою «автоматизовані системи» (АС) [4], [11].

Автоматизована інформаційна система — це взаємозв’язана сукупність даних, обладнання, програмних засобів, персоналу, стандартних процедур, які призначені для збору, обробки, розподілу, зберігання, представлення інформації згідно з вимогами, які випливають з цілей

організації. Сьогодні, у вік інформаційних технологій, практично кожна інформаційна система використовує комп'ютерні інформаційні технології (ІТ), і тому надалі під інформаційними системами будемо розуміти саме автоматизовані.

Інформаційні технології (Information and Communication Technologies) – це сукупність методів, способів, програмно-технічних засобів, інтегрованих з метою обробки, збереження, розповсюдження і використання інформації, тобто – це технології, що забезпечують та підтримують інформаційні процеси. Інформаційна технологія об'єднуються у технологічний ланцюжок, що забезпечує виконання інформаційних процесів з метою підвищення їхньої надійності та оперативності і зниження трудомісткості ходу використання інформаційного ресурсу.

CASE-технологія (Computer-Aided Software/System Engineering) являє собою сукупність методологій аналізу, проектування, розробки та супроводження складних систем програмного забезпечення (ПЗ), підтриману комплексом взаємозв'язаних засобів автоматизації ІС. CASE надає системним аналітикам, проектувальникам і програмістам інструментарій для автоматизації проектування і розроблення ПЗ [3], [5].

CASE не тільки дає змогу одержувати коректні програмні продукти, а й забезпечує технологічно правильний процес їх створення. Головна мета CASE полягає у тому, щоб відокремити проектування ПЗ від його кодування і наступних

етапів розробки, а також сховати від розробників усі деталі середовища розробки ПЗ. Основний акцент у процесі створення ПЗ припадає на етапи аналізу і проектування, на відміну від кодування.

CASE-технології широко застосовуються для багатьох типів автоматизованих систем ПЗ, але найбільше розповсюдження одержали при розробці автоматизованих систем різноманітного спрямування, в тому числі – і САПР.

Крім автоматизації структурних методологій і можливості застосування сучасних методів системної і програмної інженерії, CASE-засоби мають такі переваги:

- підвищують якість створюваного ПЗ завдяки використанню засобів автоматичного контролю, зокрема контролю проекту;

- підтримують створення прототипу майбутньої системи, що дає змогу на ранніх етапах оцінити очікуваний результат;

- прискорюють процес проектування і розроблення;

- звільняють розробника від рутинної роботи, надаючи йому можливість зосередитися на творчій частині розробки;

- підтримують розвиток і супроводження розробки;

- забезпечують технології повторного використання компонентів розробки.

2. Життєвий цикл програмного забезпечення

Методології створення програмного забезпечення тісно пов’язані з поняттям його життєвого циклу (ЖЦ). Життєвий цикл являє собою модель створення і супроводження ПЗ, що

відображає різні його стани, — від моменту виникнення ідеї створення до моменту виходу з експлуатації [4].

Типовий ЖЦ ПЗ включає такі основні етапи:

- аналіз вимог;
- проектування;
- кодування (програмування);
- тестування і налагоджування;
- експлуатація і супроводження.

ЖЦ створюється відповідно до принципу низхідного проектування і має ітераційний характер: реалізовані етапи циклічно повторюються з метою врахування змінення вимог і зовнішніх умов; введення метричних, топологічних, конструкторсько-технологічних обмежень, тощо. Результат виконання кожного етапу ЖЦ — певний набір документів і технічних рішень, які є вхідними для наступних етапів. Наприкінці кожного етапу проводиться верифікація згенерованих документів і рішень з метою перевірки їх відповідності вхідним.

Під впливом CASE-технології концепція життєвого циклу ПЗ зазнала певних змін. Ці зміни, пов’язані з автоматизацією робіт на кожному етапі, найвідчутніше вплинули на фази аналізу і проектування. На рисунку 1 наведено просту модель ЖЦ і відповідну CASE-модель, в якій традиційну фазу системного аналізу замінює фаза прототипування. Слід зазначити, що з усіх етапів ЖЦ найкраще автоматизуються етапи контролю проекту і кодогенерації.

Рисунок 1 - Моделі життєвого циклу ПЗ

3. Характеристика, склад і функціональні можливості CASE-засобів

CASE-засоби здійснюють автоматизовану підтримку робіт на всіх етапах ЖЦ ПЗ. У процесі створення і редагування проекту вони забезпечують роботу користувача в інтерактивному режимі з графічними моделями, підтримують організацію проекту у вигляді ієрархії рівнів абстракції, контролюють відповідність компонентів програмної системи .

До CASE-засобів відносять здебільшого будь-який програмний засіб, що забезпечує автоматичну допомогу в процесі розроблення ПЗ, супроводження його, а також під час управління проектом. Характерними властивостями сучасних CASE-засобів є [3], [5]:

1. Застосування потужної графіки для представлення і документування систем ПЗ, а також для поліпшення інтерфейсу користувача.

2. Використання комп'ютерного сховища, або репозиторію (repository) — бази даних CASE, в якій зберігається вся проектна інформація. Ця інформація є основою для автоматичного створення ПЗ і повторного використання його у майбутніх системах.

3. Інтеграція інформації та інструментальних засобів, що дає змогу керувати всім процесом проектування і розроблення ПЗ, використовуючи засоби планування проекту. Відокремлюють такі види інтеграції:

- інтеграція даних;
- інтеграція управління та контролю;
- інтеграція подання (зображення).

Інтеграція даних забезпечується засобами репозиторію, який містить всі дані про програмний проект і про зв'язки між цими даними, забезпечуючи можливість наскрізної навігації по проекту.

Інтеграція управління та контролю забезпечується можливостями CASE-засобів повідомляти один одного про деякі події, що відбуваються під час роботи. Ці можливості дають змогу активізувати з одного інструментального засобу інші, разом використовувати деякі функції, тощо. Інтеграція управління має забезпечувати такі типи комунікацій: між інструментальними засобами; між інструментальним засобом і сервісною функцією; між сервісними функціями.

Інтеграція подання (зображення) передбачає звертання до певного стандартного інтерфейсу користувача, який полегшує вивчення і застосування можливостей нових інструментальних засобів.

4. Широке застосування базових програмних засобів різного призначення (БД і СУБД, компілятори, налагоджувачі, документатори, текстові редактори, оболонки експертних систем і бази знань, мови четвертого покоління і т.д.).

5. Автоматизована або автоматична кодогенерація, призначена для виконання декількох видів генерації кодів: перетворення для одержання документації, формування БД, введення та модифікації даних, одержання виконуваних машинних кодів із специфікацій ПЗ, автоматичного складання модулів зі словників і моделей даних й повторно використовуваних програм, автоматичної конверсії файлів у нові формати.

6. Обмеження складності з метою одержання керованих компонентів системи з простою структурою і доступних для огляду і розуміння.

7. Доступність для різних категорій користувачів.

8. Ефективність використання і рентабельність.

9. Гнучкість, яка забезпечує здатність до адаптації за зміни вимог і цілей проекту.

Структура кожної CASE-системи має забезпечувати скоординовану взаємодію всіх її компонентів між собою і з користувачем. Велику роль при цьому відіграють можливості менеджерів задач і повідомлень, що погоджують роботу різних

інструментальних засобів. Гнучкий інтерфейс користувача покликаний забезпечувати єдиний доступ користувачів до різних компонентів та їх функцій.

Інтегрований CASE-пакет містить множину засобів, що належать до чотирьох основних груп:

1. Засоби централізованого зберігання протягом ЖЦ усієї інформації щодо проектованого ПЗ. Це так званий репозиторій, який є основою CASE-пакета. Відповідна БД має бути здатна підтримувати велику систему описів і характеристик й передбачати надійні заходи щодо захисту від помилок і втрат інформації. Репозиторій має забезпечувати:

- зберігання опису структури програми і всіх компонентів останньої;

- режим нагромадження при введенні описів об’єктів;

- поширення дії нового або скоригованого опису на інформаційний простір усього проекту;

- синхронізацію надходження інформації від різних користувачів;

- зберігання версій проекту та окремих його компонентів;

- управління складними конфігураціями і збиранням версій при побудові великих прикладних програм;

- можливості спільної роботи інструментальних засобів від різних виробників;

- контроль інформації на коректність, повноту та обґрунтованість.

2. Засоби введення, призначені для введення даних у репозиторій, а також для організації взаємодії з CASE-пакетом.

Ці засоби мають підтримувати різні методології і використовуватися протягом ЖЦ багатьма категоріями користувачів: аналітиками, проектувальниками, інженерами, адміністраторами, тощо.

3. Засоби аналізу, проектування і розроблення, що мають підтримувати планування та аналіз різноманітних описів, а також перетворення їх у процесі розроблення.

4. Засоби виведення, використовувані для документування, управління проектом і кодової генерації.

Усі наведені компоненти разом мають забезпечувати розв’язання таких функціональних завдань:

- підтримка графічних моделей;
- контроль проектної інформації;
- організація і підтримка репозиторію;
- підтримка процесу проектування і розроблення.

3.1 Підтримка графічних моделей

У процесах аналізу і проектування ПЗ одним з основних засобів відображення структури компонентів програмних систем є графічні моделі. Головними їх перевагами порівняно із словесними описами є простота і компактність, а також легкість сприйняття. Для представлення різних аспектів концептуальної моделі системи використовуються чотири типи діаграм: діаграми функціонального проектування, до яких належать DFD-діаграми потоків даних; діаграми моделювання даних (ERD-діаграми «сутність—зв’язок»); діаграми

моделювання поводження (як правило, STD-діаграми переходів станів) і структурні карти (схеми) SC.

На діаграмах потоків даних (Data Flow Diagram, DFD) відображаються потоки даних, процеси перетворення вхідних потоків на вихідні; сховища інформації, джерела і споживачі інформації, зовнішні щодо системи. Кожний з процесів може бути поданий діаграмою нижчого рівня. Надалі ці діаграми є підґрунтям для формування структури ПЗ на розробку.

На діаграмах «сутність—зв'язок» (Entity-Relationship Diagrams, ERD) мають бути показані так звані сутності (інформаційні об'єкти, що становлять інтерес з погляду наступного зберігання) і зв'язки між сутностями з відображенням характеру зв'язку. Ці діаграми є основою для проектування баз даних.

На діаграмах переходів станів (State Transition Diagram, STD) відображуються стани, в яких може перебувати система, і можливі переходи з одного стану до іншого. Згідно з діаграмою проектується інтерфейс користувача.

Структурні карти (Structure Chart, SC) слугують для відображення архітектури системи у вигляді сукупності програмних модулів і зв'язків між ними, а також даних, що передаються від одного модуля до іншого.

Засобом створення і модифікації діаграм зазначених типів є спеціальні графічні редактори (діаграмери), використовувані на етапах аналізу вимог і проектування специфікацій. Функціональні можливості сучасних діаграмерів передбачають:

- створення ієрархічно пов’язаних діаграм, в яких комбінуються графічні та текстові об’єкти;

- створення окремих об’єктів, а також груп об’єктів, і можливості редагування їх (вирівнювання, копіювання, переміщення, масштабування);

- зберігання зв’язків між об’єктами при маніпулюванні ними;

- автоматичний контроль помилок і т. ін.

Діаграмери надають зручне середовище для графічного моделювання. Одержані діаграми забезпечують стандартне подання структури системи, характеристик її елементів і функціональних зв’язків між ними.

3.2 Контроль проектної інформації

У процесі створення проектних моделей важливо організувати своєчасний контроль проектної інформації і виправлення помилок. Основну увагу при цьому слід приділити початковим етапам ЖЦ ПЗ. Дослідження зарубіжних фірм — виробників CASE виявили, що, по-перше, за традиційного підходу до створення ПЗ помилок проектування припускаються вдвічі більше, ніж помилок кодування, і, по-друге, помилки проектування в 100 разів важче виявити на етапі супроводження ПЗ, ніж на етапах аналізу вимог і проектування специфікацій.

CASE-засоби здатні забезпечити автоматичну верифікацію і контроль проекту на повноту та обґрунтованість

на ранніх етапах ЖЦ. Діаграмери і верифікатори здійснюють такі типи контролю:

1. Контроль синтаксису діаграм і типів їх елементів. За контролю цього виду перевіряється, чи додержано відповідності діаграм їх правилам побудови, а також реального заповнення елементів діаграм типам елементів.

2. Контроль повноти та обґрунтованості діаграм передбачає, що всі елементи діаграм мають бути ідентифіковані й відображені в репозиторії. Наприклад, для DFD контролюються неіменовані і незв’язані потоки даних, процеси і сховища даних.

3. Контроль декомпозиції функцій включає частковий семантичний контроль та оцінювання якості ПЗ на підставі метрик оцінки якості та встановлення відповідності останньої вимогам до якості, що надходять на вхід етапу аналізу.

4. Наскрізний контроль діаграм одного або різних типів з метою перевірки погодженості за рівнями — вертикальне і горизонтальне балансування діаграм. Вертикальне балансування застосовується для діаграм одного типу і спрямоване на виявлення незбалансованих потоків даних у діаграмі, що деталізується, та в діаграмі, що її деталізує. Горизонтальне балансування виявляє невідповідності між DFD, ERD, STD, словниками даних і мініспецифікаціями процесів. Наприклад, при балансуванні DFD—ERD контролюється відповідність кожного сховища даних на DFD сутності або відношенню на ERD.

3.3 Організація та підтримка репозиторію

Засоби організації і ведення репозиторію мають забезпечувати основні функції останнього і задовольняти вимоги щодо його вмісту. Основними функціями репозиторію є зберігання, доступ, оновлення, аналіз і візуалізація всієї інформації за проектом ПЗ. Вміст репозиторію становлять об’єкти різних типів і відношення між їх компонентами, а також правила використання та обробки цих компонентів.

У репозиторії може зберігатися понад 100 типів об’єктів, включаючи структурні діаграми, екранні форми, проекти звітів, описи даних, логіку обробки, моделі організації, вихідні коди, елементи даних і т.д.

Інформаційні об’єкти в репозиторії описуються їх властивостями: ідентифікатор, імена-синоніми, тип, текстовий опис, компоненти, файл-сховище, область значень. Зберігаються також усі відношення з іншими об’єктами (відношення входження, перехресні посилання), правила формування і редагування об’єкта, а також контрольна інформація щодо часу створення його або оновлення, номера версії, належності до певного проекту і т.д.

Засоби підтримки репозиторію мають забезпечувати функції автоматизованого формування документів за проектом. Основними типами звітів є:

- звіти, які включають зведення потоків даних, об’єктів, описи модулів, плани тестування підпрограм і т.д.;

- звіти з перехресних посилань, зокрема - дані про виклики модулів, доступ розробників до об’єктів, маршрути руху даних;

- звіти з результатів аналізу, включаючи зведення балансування діаграм, результати аналізу структури проекту;

- звіти з декомпозиції об’єктів, включаючи таблиці ієрархії всіх об’єктів моделі.

3.4 Підтримка процесу проектування і розроблення

Підтримка проектування і розроблення ПЗ означає підтримку аналізу вимог і проектування специфікацій, прототипування, підтримку структурних методологій, автоматичну кодогенерацію. До складу CASE-пакета входять засоби визначення системних вимог і моделювання очікуваного поводження системи. Для побудови прототипів використовуються генератори меню екранів і звітів, мови четвертого покоління (4GL), при цьому прототип дає можливість моделювати основні функції системи. Мови специфікацій забезпечують ітеративний процес визначення і виконання специфікацій з подальшим коригуванням. Підтримка структурних методологій здійснюється завдяки побудові структурних діаграм різних типів, генерації специфікацій для деталізації функціональних блоків у діаграмах і структур даних.

Засоби кодогенерації включають засоби генерації каркасу ПЗ і засоби генерації повного продукту. Каркас ПЗ включає опис потоків управління ПЗ, а також коди для БД, файлів,

екранів і звітів. Решта ПЗ при цьому кодується вручну. У разі генерації повного продукту з проектних специфікацій генерується повна програма разом із супровідною документацією. Найчастіше в CASE-пакетах забезпечується кодування на мовах COBOL, C і ADA.

КЛАСИФІКАЦІЯ CASE-ЗАСОБІВ

1 Вступ

CASE-засоби класифікують за різними ознаками: за типами, категоріями і рівнями.

2 Класифікація CASE-засобів за типами

Класифікація за типами розподіляє CASE-засоби за їх функціональним призначенням на такі групи [3], [5]: засоби аналізу та проектування; засоби проектування баз даних; засоби програмування; засоби супроводження і реінженерингу;

засоби міграції. Розглянемо кожний тип з вищеозначеного переліку.

2.1 Засоби аналізу та проектування

Засоби аналізу та проектування призначені для підтримки визначення системних вимог, створення специфікацій компонентів системи, проектування системи. В результаті формуються архітектура системи і детальний проект, розроблений до рівня алгоритмів і структур даних. До цієї групи належать пакети CASE. Аналітик (Ейтекс), The Developer (ASYST Technologies), POSE (Computer Systems Advisers), ProKit*Workbench (McDonnell Douglas), Excelerator (Index Technology), Design-Aid (Nastec), Design Machine (Optima), MicroStep (Meta Systems), vsDesigner (Visual Software), Analist/Designer (Yourdon), Design/IDEF (Meta Software), BPWin (Logic Works), SELECT (Select Software Tools), System Architect (Popkin Software & Systems), Westmount I-CASE Yourdon (Westmount Technology B.V. & CADRE Technologies), CASE/4/0 (micro TOOL GmbH).

2.2 Засоби проектування баз даних

Засоби проектування баз даних — забезпечують створення інфологічної та даталогічної моделей БД, нормалізацію відношень та автоматичну генерацію схем БД і описів файлів на рівні програмного коду. До цієї групи належать ERWin (Logic Works), Chen Toolkit (Chen &

Associates), S-Designor (SDP), Designer/2000 (Oracle), Silverrun (Computer Systems Advisers).

Проектування даних пов’язане з багаторівневим їх поданням: зовнішнім, інфологічним, даталогічним, внутрішнім рівнями [3], [5].

Зовнішній рівень являє собою вимоги до даних з боку користувачів і прикладних програм. Вимоги користувачів до зовнішнього подання охоплюють сукупність даних, які потрібні для виконання запитів користувачів Вимоги з боку прикладних програм до зовнішнього рівня подання даних - це перелік даних з описом їх взаємозв’язків, як необхідні для реалізації певних функціональних задач. Зовнішній рівень являє собою, як правило, словесний опис даних та їх взаємозв’язків і відбиває інформаційні потреби користувачів і прикладних програм. Іноді для опису зовнішнього рівня використовуються матричні або інші формалізовані методи. Опис зовнішнього рівня не виключає наявності дублювання, надлишковості, неузгодженості, тощо.

Для того щоб спроектувати зовнішню модель БД, необхідно виконати обстеження ПЗ, вивчити систему вхідної і вихідної документації, дослідити й вивчити всі функціональні обов’язки майбутніх користувачів БД.

Американський комітет CODASYL пропонує три рівні: зовнішній, концептуальний, внутрішній. Іноді для зручності проектування вводять допоміжний рівень (проміжний), який називають інфологічним. Він може бути й самостійним або функціонувати, як складова зовнішнього рівня. [3], [5]

Інтеграція всіх зовнішніх зображень виконується на інфологічному рівні. На цьому рівні формується інфологічна (канонічна) модель даних, яка не є простою сумою зовнішніх зображень даних. Інфологічний рівень являє собою інформаційно-логічну модель (ІЛМ) предметної області, в якій виключена надмірність даних і відображені інформаційні особливості об'єкта управління, без урахування особливостей і специфіки конкретної СУБД.

Мета інфологічного проектування - створити структуровану інформаційну модель ПЗ, для якої розроблятиметься БД. При проектуванні на інфологічному рівні створюється інформаційно-логічна модель, яка має відповідати таким вимогам:

- коректність схеми БД, тобто адекватне відображення модельованої предметної області;

- простота і зручність використання на наступних етапах проектування, тобто ІЛМ має легко відображатися в моделі БД, що підтримується відомими СУБД (мережні, ієрархічні, реляційні);

- ІЛМ має бути описана мовою, зрозумілою проектувальникам БД, програмістам, адміністратору і майбутнім користувачам БД.

Основною складовою інфологічної моделі є атрибути, які потрібно проаналізувати і деяким чином згрупувати для подальшою зберігання в БД. Сутність інфологічного моделювання полягає у виділенні інформаційних об'єктів ПЗ (файлів), які підлягають зберіганню в БД, а також визначенні

характеристик об’єктів і зв’язків між ними. Характеристиками об’єктів є атрибути.

Даталогічний (логічний, концептуальний) рівень формується з урахуванням специфіки і особливостей конкретної СУБД. На цьому рівні будується концептуальна модель даних, тобто спеціальним способом структурована модель предметної області, яка відповідає особливостям і обмеженням вибраної СУБД. Таким чином, модель логічного рівня, яка підтримується засобами конкретної СУБД, називають даталогічною. Залежно від типів моделей, які підтримуються засобами СУБД, є ієрархічні, мережні і реляційні моделі баз даних. Найпоширенішими на сучасному ринку програмних продуктів є реляційні СУБД (DBASE 111, FOXBASE, FOXPRO, CLIPPER і т. ін.) [3], [5].

Внутрішній рівень пов’язаний з фізичним розміщенням даних у пам’яті ЕОМ. На цьому рівні формується фізична модель БД, яка містить структури зберігання даних в пам’яті ЕОМ, включаючи опис форматів записів, їхнє логічне або фізичне упорядкування, розміщення за типами пристроїв, а також характеристики і шляхи доступу до даних.

Від параметрів фізичної моделі залежать такі характеристики функціонування БД: обсяг пам’яті і час реакції системи. Фізичні параметри БД можна змінювати в процесі її експлуатації (не змінюючи при цьому опису інших рівнів) з метою підвищення ефективності функціонування системи. Визначення структури масивів БД відбувається на етапах інфологічного і логічного проектування, а формування

структури - на етапі фізичного (даталогічного) проектування БД. Структура файла - це пойменована сукупність логічно взаємопов’язаних атрибутів.

Таким чином, процес проектування бази даних, що входить у загальний життєвий цикл застосування, який використовує бази даних реляційного типу, охоплює основні етапи проектування баз даних, а саме [3], [5]:

- побудова інфологічної та даталогічної моделей. Етап інфологічного моделювання можна поділити на два – концептуальний та логічний. Концептуальне проектування – створення концептуального подання бази даних, що передбачає визначення типів найважливіших сутностей і зв’язків між ними та атрибутів. Логічне проектування – перетворення концептуального уявлення на логічну структуру бази даних, зокрема проектування взаємозв’язків. Після завершення етапу побудови інфологічної моделі бази даних можна переходити до даталогічної моделі. Даталогічне проектування – ухвалення рішення про те, як логічна модель буде реалізована (за допомогою таблиць) у базі даних, що створюється з використанням вибраної СУБД.

2.3 Засоби програмування

Засоби програмування здійснюють підтримку програмування і тестування, а також автоматичну кодогенерацію зі специфікацій з одержанням повністю документованої виконуваної програми. До цієї групи належать діаграмери і засоби роботи з репозиторієм, генератори та

аналізатори кодів, генератори тестів, аналізатори покриття тестами, налагоджувачі. Основні пакети: COBOL 2/Workbench (Mikro Focus), DECASE (DEC), NETRON/CAP (Netron), APS (Sage Software).

2.4 Засоби супроводження

Засоби супроводження забезпечують управління функціонуванням системи, коригування і модифікацію, аналіз і реінжиніринг існуючої системи. До них належать документатори, аналізатори програм, засоби міграції, засоби реструктурування та реінжинірингу: Adpac CASE Tools (Adpac), Scan/COBOL та SuperStructure (Computer Data Systems), Inspector/ Recoder (Language Technology).

2.5 Засоби міграції і реінжинірингу

Засоби міграції призначені для перенесення існуючої системи в нове операційне або апаратне оточення і включають транслятори, конвертори, макрогенератори і т.д.

Засоби реінжинірингу включають [3]:

- статичні аналізатори для одержання схем системи з її кодів та оцінки впливу окремих модифікацій на всю систему;

- динамічні аналізатори типу компіляторів та інтерпретаторів із вбудованими можливостями налагоджування;

- документатори, здатні автоматично оновлювати документацію при змінюванні коду;

- редактори кодів, що при редагуванні автоматично змінюють всі структури, які передують коду (наприклад, специфікації);

- засоби доступу до специфікацій для їх модифікації і генерації нового коду;

- засоби реверсного реінжинірингу, що транслюють програмні коди в специфікації програм.

2.6 Засоби оточення

Засоби оточення включають засоби підтримки каркасів і платформ для створення, інтеграції і надання CASE-засобам товарного вигляду: Multi/Cam (AGS Management Systems), Sylva Foundry (Cadware).

Засоби управління проектом призначені для підтримки планування, контролю, керування і взаємодії у процесі розроблення та супроводження проектів: Project Workbench (Applied Business Technology).

3 Класифікація за категоріями

Класифікація за категоріями ґрунтується на визначенні рівня інтегрованості функцій у CASE-пакетах і поділяє їх на допоміжні програми (tools), пакети розробника (toolkit) та інструментальні засоби (workbench) [3], [5].

Категорія tools включає допоміжні пакети, що розв’язують нескладні автономні задачі.

Категорія toolkit характеризує інтегровані засоби підтримки одного з класів програмних задач, зорієнтована на підтримку одного етапу ЖЦ, використовує репозиторій для зберігання проектної інформації.

Категорія workbench об'єднує інтегровані програмні засоби, що організують підтримку повного ЖЦ ПЗ, включаючи аналіз вимог, проектування і програмування, використовують репозиторій, забезпечують автоматичну передачу системної інформації від одного проектувальника до іншого та між етапами розробки. Ця категорія характеризується тісним зв'язком з системними і технічними засобами, на яких workbench функціонує. Остання може розглядатися, як автоматизована робоча станція, що виконує функції автоматизації всіх або деякого набору робіт зі створення ПЗ автоматизованої системи.

4 Класифікація за рівнями

Класифікація за рівнями враховує рівень функцій, які реалізують CASE-засоби в межах ЖЦ ПЗ: верхні CASE; середні CASE; нижні CASE.

Верхні CASE інакше називають засобами комп'ютерного планування. Вони призначені для підтримки управління проектом і виконують роботи зі створення плану проекту, моделювання предметної області, аналізу сценаріїв, накопичення інформації для прийняття оптимальних рішень.

Середні CASE включають засоби підтримки етапів аналізу вимог і проектування специфікацій та структури ПЗ. Крім того,

вони забезпечують швидке документування вимог і швидке прототипування.

Нижні CASE забезпечують перетворення специфікацій на програмні коди, а також виконують функції тестування, управління конфігурацією, формування документації. Головними їх перевагами є значне зменшення тривалості розроблення програм і полегшення модифікацій ПЗ.

ЗАСОБИ СТРУКТУРНОГО АНАЛІЗУ ТА ПРОЕКТУВАННЯ СИСТЕМ

1 Вступ

2 Діаграми потоків даних

2.1 Зовнішня сутність (термінатор)

2.2 Текст

2.3 Керуючий процес та керуюче сховище

2.4 Керуючий потік

2.5 Вузол змінювання типу

3 Словник даних

4 Специфікації процесів

5 Діаграми «сутність - зв’язок»

6 Перевірка ERD на коректність

6.1 Діаграми переходів станів

6.2 Структурні карти

7 Методологія структурного аналізу проектної системи

8 Класифікація методологій структурного аналізу та проектування

1 Вступ

Серед усіх етапів ЖЦ ПЗ етап аналізу вважається найвідповідальнішим, оскільки саме на ньому виробляються фундаментальні рішення щодо системи, які уточнюватимуться на наступних етапах [3], [5].

Етап аналізу є першою фазою розробки ПЗ, на якій конкретизуються, формалізуються і документуються вимоги замовника. Список вимог до системи містить такі пункти:

- сукупність експлуатаційних умов (апаратні та програмні ресурси, зовнішні умови функціонування системи, склад людських ресурсів, роботи, пов’язані з системою);

- опис функцій, виконуваних системою;

- обмежувальні фактори, що впливають на розробку автоматизованої системи (директивні терміни закінчення етапів; ресурсні обмеження; заходи щодо захисту інформації, тощо).

Головною метою аналізу є перетворення цих неточних вимог на точні визначення. На цьому етапі виробляються:

- архітектура і функції системи, вимоги до зовнішнього середовища, розподіл функцій між ПЗ і апаратними засобами;

- інтерфейси і розподіл функцій між людиною і програмною системою;

- вимоги до компонентів програмної системи — програмних та інформаційних, зокрема до БД; вимоги до апаратних ресурсів; фізичні характеристики компонентів ПЗ та їх інтерфейси.

Структурний аналіз передбачає такий підхід до вивчення системи, який починають із загального огляду її, а потім поступово переходять до детальніших рівнів, утворюючи ієрархію описів. Головними принципами структурного аналізу є принцип розподілу на окремі частини і принцип ієрархічного впорядкування. Крім того, у цьому аналізі втілюються принципи: абстрагування, формалізації, приховування несуттєвих деталей на кожному рівні; концептуальної єдності з усіма етапами ЖЦ; повноти і контролю щодо присутності зайвих елементів; несуперечливості елементів, незалежності логічного проектування від фізичного; структурування даних; безпосереднього доступу користувача до БД.

Етап аналізу передбачає розроблення сукупності моделей, що відображують різні аспекти проблеми. Процеси і потоки інформації, що протікають між ними, відображуються діаграмами потоків даних (DFD), інформаційні потреби розроблюваної системи — діаграмами «сутність—зв’язок» (ERD), часові аспекти функціонування — діаграмами переходів станів (STD). Етап завершується формуванням попередньої архітектури системи із сукупності моделей. Описи всіх елементів моделей зосереджуються у словнику даних. В результаті одержують:

- модель ПЗ, що являє собою сукупність моделей DFD, ERD і STD;

- словник даних з описом усіх елементів моделей: потоків даних, сховищ даних, процесів (у вигляді специфікацій),

сутностей та їх атрибутів, а також зв’язків між сутностями (з описом характеру цих зв’язків), станів і переходів;

- план тестування;

- попередню архітектуру системи.

Одержана інформація називається структурною специфікацією на ПЗ і відображує вимоги до його розробки.

Структурне проектування починається з аналізу попередньої архітектури, сформованої на етапі аналізу. Аналіз ґрунтується на критеріях якості, визначених для архітектури. Архітектура системи носить логічний характер у тому розумінні, що вона не прив’язана до конкретних мов програмування, операційних систем та особливостей ЗОТ. Подальша робота полягає в урахуванні цих особливостей, завдяки чому одержують фізичну архітектуру. На цьому ж етапі проектуються БД та інтерфейс користувача. В результаті отримують:

- архітектуру системи у вигляді структурних карт, на яких відображено програмні модулі та зв’язки між ними;

- проект БД та інтерфейсу користувача;

- словник даних, що містить опис системних даних;

- специфікації модулів з описом їх призначення та особливостей;

- проект архітектурних тестів (в яких модулі розглядаються як «чорний ящик» без урахування їх внутрішньої структури).

Ця інформація передається на подальші етапи створення ПЗ.

Серед розмаїття засобів структурного аналізу і проектування, центральне місце посідають діаграми потоків даних (DFD) разом із словниками даних і специфікаціями процесів, або міні-специфікаціями, діаграми «сутність—зв'язок» (ERD) і діаграми переходів станів (STD).

На DFD зображуються зовнішні, відносно системи, джерела і стоки (адресати) даних, логічні функції (процеси) і групи елементів даних, що пов'язують одну функцію з іншою (потоки), а також сховища (нагромаджувачі) даних, до яких здійснюється доступ.

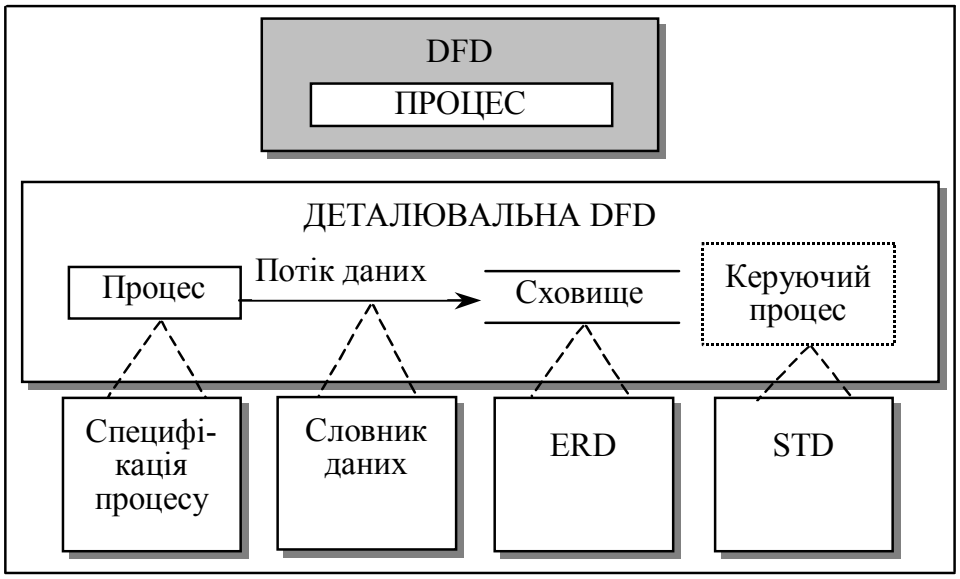


Рисунок 1 - Компоненти логічної моделі

Структури потоків даних і визначення їх компонентів зберігаються та аналізуються у словнику даних. Кожний процес може бути деталізований за допомогою DFD нижчого рівня; коли подальша деталізація стає недоцільною, логіку процесу описують за допомогою специфікації (міні-специфікації) процесу. Вміст кожного сховища також зберігають у словнику даних, структура даних сховища

розкривається за допомогою ERD. Для відображення реального часу залежну від часу поведінку системи описують за допомогою STD. Ці зв'язки між засобами наведено на рисунку 1.

2. Діаграми потоків даних

Діаграми потоків даних (DFD) є основним засобом моделювання функціональних вимог системи [3], [5]. Такі діаграми розкладають ці вимоги на функціональні компоненти (процеси) і показують зв'язки між ними за допомогою потоків і сховищ даних. Основні символи DFD у нотації Йордана наведено в таблиці 1.

Таблиця 1 - Основні символи діаграм потоків даних

Символ DFD	Зображення
Потік даних	ім'я
Процес	ім'я номер
Сховище	ім'я
Зовнішня сутність	ім'я
Вузол-предок	
Груповий вузол	
Невикористовуваний вузол	**NU**
Вузол змінювання імені	**N** ім'я 1 ім'я 2

Потоки даних є механізмами, призначеними для відображення передавання інформації (або навіть фізичних об’єктів) з однієї частини системи в іншу. Як правило, на діаграмах потоки зображуються у вигляді іменованих стрілок, орієнтація яких показує напрямок руху даних. Стрілки можуть бути двонапрямленими, якщо передана інформація після обробки повертається до свого джерела.

Процеси призначені для показу обробки даних, в результаті якої на основі вхідних потоків утворюються вихідні. Ім’я процесу відображує дію, яка у ньому відбувається, і складається з дієслова у неозначеній формі та додатка (наприклад, РОЗРАХУВАТИ ПЛОЩУ ДИСКРЕТНОГО МОНТАЖНОГО ПРОСТОРУ). Кожному процесу має бути присвоєний унікальний номер усередині діаграми. Разом із номером діаграми цей номер утворює унікальний індекс процесу в усій моделі.

Сховище (нагромаджувач) даних визначає сукупності даних, які зберігатимуться у пам’яті між процесами. Інформація зі сховища може використовуватись у будь-який час після її визначення, при цьому дані можуть вибиратись у довільному порядку. Ім’я сховища має ідентифікувати його вміст і виражається іменником. У тому разі, коли структура потоку даних відповідає структурі сховища, до якого він входить або з якого виходить, цей потік повинен мати те саме ім’я, яке необов’язково відображати на діаграмі.

2.1 Зовнішня сутність (термінатор)

Зовнішня сутність (термінатор) являє собою сутність поза контекстом системи, що є джерелом або приймачем системних даних [3], [5]. Ім’я зовнішньої сутності має містити іменник (наприклад, ЗАМОВНИК СИСТЕМИ). Передбачається, що об’єкти, представлені такими вузлами, не беруть участі в жодній обробці.

Деталізація DFD відбувається на основі процесів. Це означає, що кожний процес може розкриватися за допомогою DFD нижчого рівня. На верхньому рівні ієрархії DFD розташовується DFD спеціального виду — контекстна діаграма, що моделює систему найбільш узагальнено. Така діаграма відображує інтерфейс системи із зовнішнім середовищем, при цьому до її складу входить єдиний процес, що відбиває головну мету системи, зовнішні сутності та інформаційні потоки між процесом системи і зовнішніми сутностями. Кожний проект повинен мати тільки одну контекстну діаграму.

DFD першого рівня являє собою декомпозицію процесу, присутнього на контекстній діаграмі; DFD другого рівня деталізують процеси діаграми першого рівня і т.д. Цей процес повторюється доти, доки подальша деталізація процесів за допомогою діаграм потоків даних не втратить сенсу. Тоді процеси нижнього рівня описують за допомогою коротких (до однієї сторінки) міні-специфікацій обробки (специфікацій процесів). Номери процесів, як правило, утворюються

додаванням до номера процесу вищого рівня (наприклад, 2.2) порядкових номерів процесів у межах діаграми, що його деталізує (наприклад, 2.2.1, 2.2.2).

Інформаційні потоки можуть мати у своєму складі кілька незалежних потоків; такі потоки називаються груповими. Операції розщеплення та об'єднання потоків даних позначаються груповим вузлом (дивись таблицю.1). Для забезпечення декомпозиції даних і деяких інших сервісних можливостей, у DFD використовуються такі типи об'єктів:

- вузол-предок — надає можливість пов'язувати вхідні і вихідні потоки між процесом, що деталізується, і DFD, що його деталізує;

- груповий вузол — призначений для розщеплення та об'єднання потоків даних. Іноді він відсутній, тобто вироджується в точку злиття/розщеплення потоків на діаграмі;

- невикористовуваний вузол — застосовується для декомпозиції даних, якщо потрібні не всі елементи вхідного потоку;

- вузол змінювання імені дає змогу неоднозначно іменувати потоки з еквівалентним вмістом. Використовується тоді, коли потрібно встановити ідентичність між двома різнойменними потоками. При цьому один з них є вхідним для даного вузла, а інший — вихідним.

2.2 Текст

Текст у довільному форматі використовується в будь-якому місці діаграми.

Для відображення у DFD аспектів управління у системах реального часу використовуються спеціальні символи (таблиця 2).

Таблиця 2 - Символи управління в діаграмах потоків даних

Символ DFD	Зображення
Керуючий потік	ім’я
Керуючий процес	ім’я номер
Керуюче сховище	ім’я
Вузол змінювання типу	Т ім’я 1 ім’я 2

2.3 Керуючий процес та керуюче сховище

Керуючий процес являє собою інтерфейс між DFD і специфікаціями управління (діаграмами переходів станів), що безпосередньо моделюють і документують аспекти реального часу. Ім’я керуючого процесу вказує на зміст управлінської діяльності. Фактично керуючий процес перетворює вхідні керуючі потоки на вихідні керуючі потоки. Точний опис цього перетворення має міститись у специфікаціях управління.

Керуюче сховище містить керуючу інформацію, яка після занесення її у сховище може використовуватись у будь-який час і в будь-якому порядку. Ім’я керуючого сховища має відповідати його вмісту і бути іменником.

2.4 Керуючий потік

Керуючий потік являє собою «провідник» для управляючої інформації. Остання, як правило, подається дискретними сигналами. Його ім’я складається з іменників і прикметників. Керуючі потоки можуть як повідомляти керуючому процесу про зміни зовнішніх умов, так і виконувати команди, які він генерує. При цьому режим виконання керуючого процесу залежить від типу керуючого потоку. Використовуються такі типи керуючих потоків:

T-потік (trigger flow) — керуючий потік, що може викликати виконання процесу за допомогою однієї короткої операції. Він діє так само, як вмикач світла, єдине натиснення на який спричинює засвічування електричної лампи;

A-потік (activator flow) — керуючий потік, що може змінювати виконання окремого процесу, а саме: забезпечувати безперервне виконання процесу (підпроцесу) доти, доки потік «увімкнений», тобто плине безперервно; з «вимкненням» потоку виконання процесу (підпроцесу) закінчується. Діє, як перемикач електричної лампочки, здатний спрацьовувати на її вмикання і вимикання;

E/D-потік (enable/disable flow) — потік керування процесом, що може перемикати виконання окремого процесу. Доки керуюча інформація надходить по E-лінії, процес триває; при збудженні D-лінії — припиняється. Це аналог вимикача з двома кнопками: одна — для вмикання світла, інша — для вимикання. Можна використовувати три типи таких потоків: E-потік, D-потік, E/D-потік.

2.5 Вузол змінювання типу

Вузол змінювання типу використовується для змінювання типу керуючого потоку, а також для заміни типу потоку даних на тип керуючого потоку і навпаки. При цьому вміст потоку не змінюється. Побудову ієрархії діаграм потоків даних (DFD) корисно здійснювати у такій послідовності:

- вивчення множини вимог і розподіл їх на декілька основних функціональних груп;

- ідентифікація зовнішніх об’єктів, з якими має бути зв’язана система, і основних видів інформації, що циркулює між об’єктами та системою;

- розроблення попереднього варіанта контекстної діаграми, на якій основні функціональні групи подаються процесами, зовнішні об’єкти — зовнішніми сутностями, основні види інформації — потоками даних між процесами і зовнішніми сутностями;

- аналіз попередньої контекстної діаграми і внесення в неї змін за результатами аналізу;

- побудова контекстної діаграми об’єднанням усіх процесів попередньої діаграми в один процес, а також групуванням потоків даних;

- формування DFD першого рівня на базі процесів попередньої контекстної діаграми;

- перевірка основних вимог за DFD поточного рівня і внесення змін (за потреби). Розбиття вимог на більш детальні та ідентифікація процесів або специфікацій процесів, які відповідають цим вимогам;

- додавання визначень нових потоків до словника даних при появі їх на діаграмі;

- декомпозиція кожного процесу поточної DFD за допомогою детальної діаграми або специфікації процесу в разі, якщо функцію процесу складно або неможливо подати комбінацією процесів;

- після побудови чергових двох-трьох рівнів моделі проведення ревізії з метою перевірки коректності та підвищення зрозумілості моделі.

Якщо діаграма містить процеси, не описані специфікаціями, перехід до перевірки основних вимог за DFD поточного рівня і внесення змін (за потреби). Прикладом побудови контекстної DFD може бути діаграма процесу автоматизації проектної процедури трасування з'єднань на основі наявної схеми електричної принципової з послідуючим виготовлення комплекту конструкторських документів та складенням акта приймання цього комплекту замовником (рисунок 2).

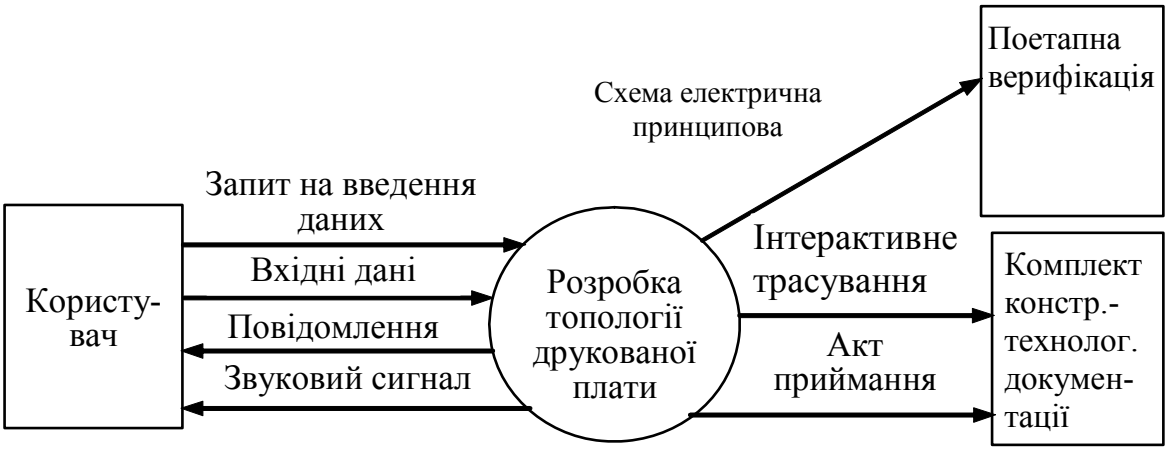


Рисунок 2 - Контекстна діаграма

3. Словник даних

Побудова діаграм потоків даних (DFD) обов'язково має супроводжуватися формуванням описів усіх даних, що циркулюють у системі. Для цього використовуються текстові засоби моделювання — словники даних [3], [5]. Словник даних являє собою певним чином організований список усіх елементів даних системи з точними визначеннями їх. Для визначення даних застосовуються такі види описів:

- опис вмісту потоків і сховищ, зображених на DFD;

- опис композиції агрегатів даних (комплексних даних) у складі потоків і сховищ;

- уточнення значень і областей дії елементарних фрагментів інформації у потоках даних і сховищах;

- опис відношень між сховищами.

Опис потоку даних у словнику містить ім'я потоку, його тип і атрибути. Опис складається з набору статей, кожна з яких починається із символу «@» і ключового слова — заголовка статті.

Інформація за типом потоку визначає:

- прості (елементарні) та групові (комплексні) потоки;

- внутрішні (що існують усередині системи) і зовнішні (що пов'язують систему з іншими системами) потоки;

- потоки даних або керуючі потоки;

- неперервні (що набувають будь-яких значень з певного діапазону) або дискретні (що набувають лише певних значень) потоки.

Атрибути потоку даних можуть включати:

- імена-синоніми потоку даних у відповідності до вузлів змінювання імені;

- БНФ (визначення у формі Бекуса-Наура) — визначення для групових потоків;

- одиниці вимірювання потоку;

- діапазон значень для неперервних потоків і список значень та їх сенс — для дискретних потоків;

- список номерів діаграм різних типів, до яких може належати потік;

- список групових потоків, до якого входить даний потік (як елемент БНФ - визначення);

- коментар з додатковою інформацією.

Для формального опису,поділу та об’єднання потоків може бути використане БНФ-визначення. При цьому важливо, щоб кожний компонент потоку-предка був іменованим. Об’єднуючи підпотоки, не обов’язково вилучати спільні компоненти, а в разі поділу потоків, підпотоки можуть мати однакові компоненти. БНФ-визначення зберігається у словнику даних у так званій БНФ-статті. БНФ-стаття використовується для опису компонентів даних у потоках даних та у сховищах і має такий синтаксис:

@ БНФ = <простий оператор> ! <БНФ-вираз> ,

де <простий оператор> — це текстовий опис, узятий в «/», а <БНФ-вираз> є виразом у формі Бекуса-Наура, який допускає операції відношення, що тлумачаться так:

= — «композиція з»;

+ — «І»;

[!] — «АБО»;

( ) — компонент у дужках необов’язковий;

{ } — повторення компонента в дужках;

« » — літерал.

При зазначенні кількості повторень, число перед дужкою вказує нижню межу повторення, а число після дужки — верхню. Наприклад:

3{сигнал}7 — від 3 до 7 сигналів.

Приклад опису потоку даних за допомогою БНФ:

@ Ім’я = двійкова цифра;

@ ТИП = дискретний потік;

@ БНФ = [«0» ! «1»].

4. Специфікації процесів

Специфікація процесу (СП) являє собою короткий (до однієї сторінки) опис процесу, створюваний тоді, коли подальша деталізація процесу за допомогою DFD стає недоцільною [3], [5]. Сукупність усіх СП є повною специфікацією системи. За своєю суттю СП є описом алгоритму задачі, виконуваної процесом, містить номер і/або ім’я процесу, перелік вхідних і вихідних даних, опис тіла процесу, тобто - алгоритму чи процедури, що перетворює вхідні потоки даних на вихідні.

Специфікація процесу починається з опису вхідних і вихідних даних, після чого вказується ключове слово, наприклад @ СПЕЦПРОЦ:

@ ВХІД = <ім’я елемента даних>

@ ВИХІД = <ім’я елемента даних>

@ СПЕЦПРОЦ <номер і/або ім’я процесу> ,

де <ім’я елемента даних> — відповідне ім’я із словника даних.

Якщо елемент даних одночасно є вхідним і вихідним для процесу, він може бути описаний двічі за допомогою @ ВХІД і @ ВИХІД або одноразово за допомогою @ ВХІД/ВИХІД.

Для опису тіла процесу існує багато методів, серед яких можна вирізнити структуровану природну мову або псевдокод, візуальні мови проектування (FLOW-форми та діаграми Нассі—Шнейдермана) та формальні комп’ютерні мови.

5. Діаграми «сутність—зв’язок»

Діаграми «сутність—зв’язок» (ERD) призначені для моделювання представлення даних у програмних системах. За їх допомогою ідентифікуються інформаційні об’єкти (сутності) системи, їх властивості (атрибути) і відношення між об’єктами (зв’язки) [3], [5].

Нотація ERD була розроблена Ченом і набула розвитку у працях Баркера, Бахмана та інших учених. Символи ERD у нотації Чена наведено на рисунку 3.

Сутність являє собою множину примірників деякого об’єкта (реального чи абстрактного), які мають спільні характеристики або атрибути. Відношення визначає взаємостосунки двох чи більше сутностей. Ім’я відношення містить дієслово (наприклад, ВОЛОДІЄ, ВИЗНАЧАЄ, МОЖЕ МАТИ).

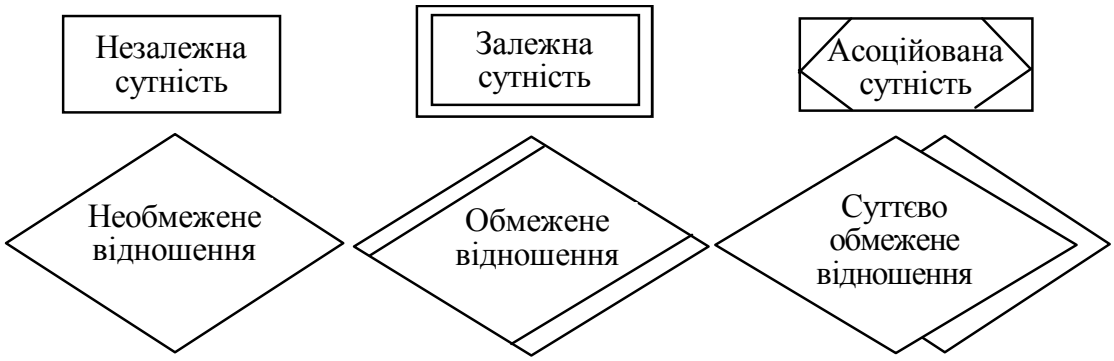


Рисунок 3 - Символи ERD у нотації Чена

Незалежна сутність представляє незалежні об’єкти, завжди присутні в системі, які можуть не мати відношень з іншими сутностями. Залежна сутність представляє дані, залежні від інших сутностей. Асоційована сутність представляє об’єкти, асоційовані з двома і більше сутностями.

Необмежене (обов’язкове) відношення являє собою безумовне відношення, тобто відношення, що існує доти, доки існують сутності, між якими воно встановлене. Обмежене (необов’язкове) відношення являє собою умовне відношення між сутностями. Суттєво обмежене відношення використовується, якщо відповідні сутності є взаємозалежними.

Між відношеннями та сутностями у діаграмі встановлюються зв’язки, напрямлені від відношення до сутності. Значення зв’язку характеризує кількість примірників сутності, що беруть участь у кожному відношенні («0 або 1», «0 і більше», «1», «1 і більше», «p:q»). Пара значень зв’язків відношення визначає тип даного відношення. Основні типи

відношень такі: 1*1 (один до одного), 1*n (один до багатьох), n*m (багато до багатьох).

Фрагмент ER-діаграми, що відображує відношення між сутностями комплексної САПР розробки та виготовлення на автоматизованій лінії друкованих плат: наявність ЕРЕ на спеціалізованому складі підготовки елементів до монтажу технологічна операція формування виводів ЕРЕ) на спеціалізованому складі, наведено на рисунку 4.

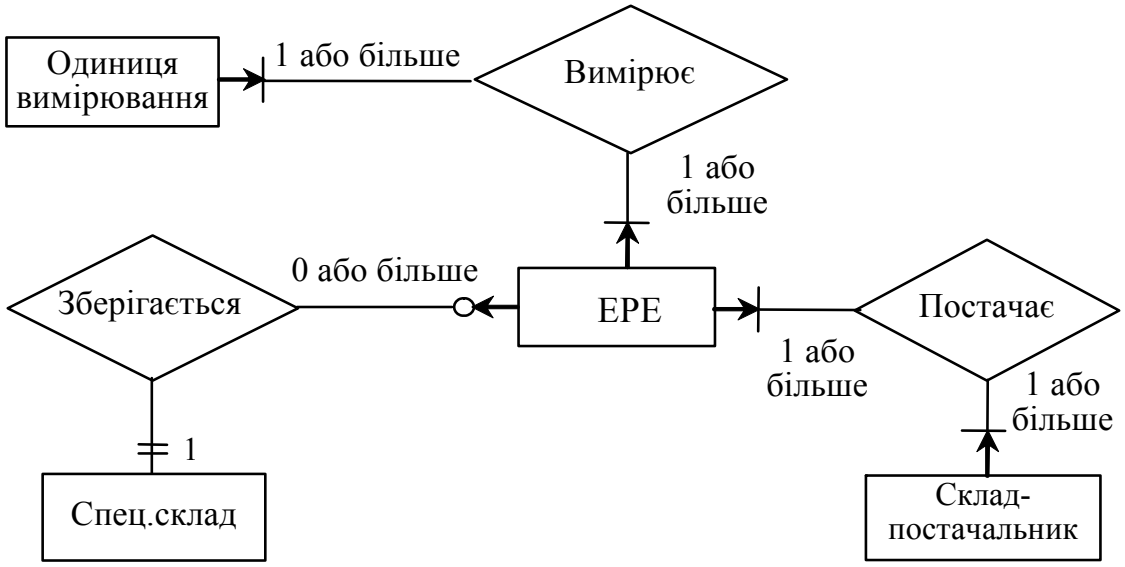


Рисунок 4 - Фрагмент ER-діаграми в нотації Чена

Сутності деталізуються за допомогою діаграм атрибутів. Останні містять у собі сутність, що деталізується, атрибути сутності, домени, які описують області значень атрибутів, та зв’язки. Для позначення ключового атрибута його ім’я підкреслюється.

Інший спосіб конкретизації сутностей — їх категоризація, за якої сутність поділяється на сутності-категорії, що можуть

мати спільні атрибути і/або відношення, а також власні атрибути і відношення. Сутність, що поділяється на категорії, називається спільною сутністю. Процес поділу на категорії відображується на діаграмах категоризації, які крім спільної сутності і сутностей категорій включають спеціальний вузол — дискримінатор, що описує спосіб декомпозиції сутностей (повне і неповне, обов'язкове і необов'язкове входження).

В інших нотаціях ERD зв'язки між об'єктами можуть зображуватися поодинокими і подвоєними стрілками. Необов'язковий зв'язок може позначатися пунктирною лінією, тип співвідношення — літерою поряд зі зв'язком і т.д.

Основні етапи розроблення ERD такі:

1. Ідентифікація атрибутів, агрегація сутностей, а також первинних і альтернативних ключів.

2. Нормалізація сутностей, тобто поділ груп атрибутів, усунення неповної функціональної залежності від ключів, транзитивної залежності між неключовими атрибутами, множинної залежності.

3. Ідентифікація відношень між сутностями та їх типами

6 Перевірка ERD на коректність

Зауважимо, що етап 2 передбачає використання теорії нормалізації Е.Ф.Кодда. Цей учений встановив існування трьох типів нормалізованих схем — першої, другої і третьої нормальної форм (відповідно, 1НФ, 2НФ, 3НФ) зі спадним ступенем складності [3], [5].

6.1 Діаграми переходів станів

Діаграми переходів станів (STD) відносяться до групи специфікацій управління, яка призначена для моделювання і документування аспектів системи, пов’язаних із часом або реакцією на події. Вони дають змогу деталізувати керуючі процеси і описати взаємодію між вхідними і вихідними керуючими потоками. STD подають процес функціонування системи, як послідовність переходів з одного стану до іншого.

До складу STD входять такі структурні одиниці:

- стан — може визначатися як стійкі внутрішні умови системи. При моделюванні поводження системи в кожному певному стані достатньо знати попередню її історію і поточні вхідні події, щоб визначити наступний її стан. Ім’я стану має відображати реальну ситуацію, в якій перебуває система;

- початковий стан — вузол STD, який є стартовою точкою для початкового системного переходу. STD має тільки один початковий стан, що відповідає стану системи після інсталяції її перед початком обробки, а також будь-яку скінченну кількість кінцевих станів;

- перехід — визначає переміщення системи з одного стану до іншого. Ім’я переходу ідентифікує деяку подію (умову), що є причиною переходу і керує ним. Ця умова може бути пов’язана з керуючим потоком, що виникає всередині системи або надходить ззовні. Якщо в умові бере участь керуючий потік керуючого процесу-предка, то ім’я керуючого потоку береться в лапки (наприклад, ПЕРЕМИКАЧ = «КОРЕКТНИЙ ПАРОЛЬ»).

Окрім умови, з переходом може пов’язуватися дія або послідовність дій, які виконуються при переході. Якщо ця дія стосується вибору вихідного керуючого потоку, то ім’я цього потоку має бути також взяте в лапки (наприклад, «ВИБІР ФУНКЦІЇ» = TRUE, де ВИБІР ФУНКЦІЇ — вихідний потік).

Крім того, при специфікації керуючих потоків типу A-, T-, E/D-ім’я процесу, що запускається або перемикається, також береться в лапки. Наприклад:

A: «ОДЕРЖАТИ ПАРОЛЬ» — активізувати процес ОДЕРЖАТИ ПАРОЛЬ.

На STD стани подаються вузлами, а переходи — орієнтованими дугами (рисунок 5). Умови або стимулюючі події, ідентифікуються іменем переходу. Дії або відгуки на події, прив’язуються до переходу і записуються під відповідною умовою. Початковий стан на діаграмі повинен мати вхідний перехід від стартового вузла, який може не відображатися на схемі або подаватися у вигляді невеликого квадрату.

При побудові STD рекомендується будувати першу STD на якомога вищому рівні узагальнення DFD, а далі, по змозі, деталізувати її. При побудові STD слід контролювати визначення всіх станів та унікальність їх імен, досяжність станів, наявність виходів у всіх станів, наявність реакції системи на всі умови, відображення всіх вхідних і вихідних потоків керуючого процесу в умовах і діях на STD.

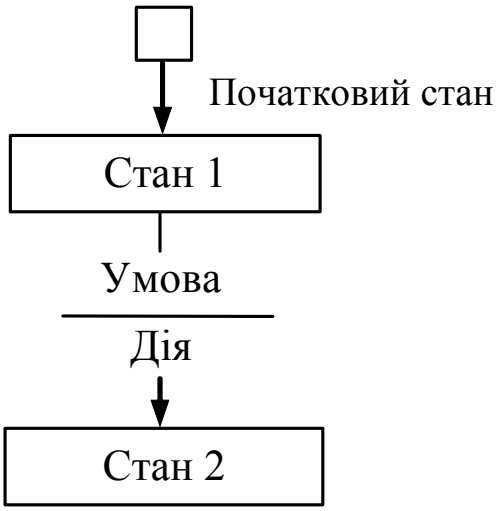


Рисунок 5 - Символи STD

6.2 Структурні карти

У попередніх підрозділах були розглянуті засоби підтримки структурного системного аналізу, застосовувані для побудови логічної моделі системи, тобто - моделі вимог. Логічна модель об'єднує множину взаємопов'язаних діаграм (DFD, ERD, STD), специфікацій процесів, текстів і словника даних. Разом вони показують, що саме робитиме система, але не пояснюють, завдяки чому це відбуватиметься [3], [5].

На етапі проектування аналізується і поглиблюється логічна модель з метою зведення її проектних рішень до рівня реалізації (без технічних подробиць). У результаті формується фізична модель, яка є розширенням логічної моделі і містить модель фізичної архітектури системи у вигляді структурної карти (схеми).

Техніка структурних карт використовується на фазі проектування для побудови програмної структури системи.

Найчастіше застосовуються дві техніки: структурні карти Константайна, призначені для відображення відношень між модулями, і структурні карти Джексона, призначені для подання внутрішньої структури модулів.

7 Методології структурного аналізу і проектування систем

Методологія структурного аналізу і проектування ПЗ являє собою сукупність правил і методів для визначення кроків роботи при створенні ПЗ та їх послідовності, для вибору засобів і призначення операцій, а також для оцінювання проекту ПЗ.

Останнім часом серед усіх методологій структурного аналізу і проектування найбільш поширені методології SADT (Structured Analysys and Design Technique), структурного системного аналізу Гейна—Сарсона, структурного аналізу та проектування Йордана—Де Марко, розвитку систем Джексона, розвитку структурних систем Варньє—Орра, аналізу і проектування систем реального часу Уорда—Меллора і Хатлі, інформаційного моделювання Мартіна [3], [5].

Структурні методології регламентують фази аналізу вимог і проектування специфікацій і надають точні вказівки для здійснення переходу від специфікацій вимог до проекту архітектури програмної системи та проектних специфікацій, пропонують методику трансляції проектних специфікацій у модель реалізації, що далі буде використана при кодогенерації. Кодогенерація виконується в рамках кодових стандартів, які

визначають формат заголовків підпрограм, ступінчастий вигляд вкладених блоків, номенклатуру для специфікації змінних та імен підпрограм і т.д.

Методології структурного системного аналізу і проектування використовують багато різноманітних засобів і діаграмних технік. Основні з них такі:

- діаграми потоків даних у нотації Йордана—Де Марко або Гейна—Сарсона, що забезпечують аналіз вимог і функціональне проектування інформаційних систем;

- методики Хатлі та Уорда—Меллора проектування систем реального часу, що ґрунтуються на діаграмах переходів станів, таблицях і деревах вирішень, картах і схемах потоків управління;

- діаграми «сутність—зв'язок» (у нотації Чена чи Баркера) або дужкові діаграми Варньє—Орра для проектування структур даних, схем БД, форматів файлів;

- структурні карти Джексона і Константайна для проектування міжмодульної структури програм і внутрішньої структури модулів

8 Класифікація методологій структурного аналізу та проектування

Структурні методології аналізу та проектування залежно від ознаки класифікації поділяються на такі групи:

1. За належністю до шкіл проектування систем — на інжиніринг ПЗ (Software Engineering, SE) та інформаційний інжиніринг (Information Engineering, IE). Методологія SE

втілює низхідний поетапний підхід до розроблення ПЗ, згідно з яким, визначаються і поступово деталізуються функції системи із створенням ієрархічно організованої модульної програми. Методологія ІЕ репрезентує порівняно нову школу, що вивчає створення інформаційних систем і охоплює етапи вищого рівня, такі, як стратегічне планування.

2. За порядком побудови моделі — на процедурно-орієнтовані, інформаційно-орієнтовані та орієнтовані на дані.

Процедурно-орієнтований підхід при проектуванні ґрунтується на тому, що процедури обробки вважаються першорядними, а структури даних — другорядними; вимоги до даних визначаються в результаті аналізу функціональних вимог.

3. За інформаційно-орієнтованого підходу основна увага спрямовується на вивчення входів і виходів системи та розроблення структур даних, після якого проектують функціональні процедури.

4. Підхід, орієнтований на дані, відрізняється від інформаційно-орієнтованого підходу тим, що дає змогу працювати лише з ієрархічними структурами даних.

5. За типом цільових систем — на методології для систем реального часу (СРЧ) і для інформаційних систем (ІС). Ці типи методологій спрямовані на підтримку особливостей відповідних систем.

ВИКОРИСТАННЯ ЗАСОБІВ MS EXCEL ЯК OLAP-КЛІЄНТА ДЛЯ ОПЕРАТИВНОГО АНАЛІЗУ ДАНИХ

Вступ

Технологія оперативного аналітичного оброблення даних OLAP виникла в зв'язку із необхідністю аналізу даних, накопичуваних в інформаційних системах у результаті розв'язування облікових задач. Ця технологія орієнтована на побудову звітів у різноманітних напрямах, із різним ступенем проникнення в деталі, на прогнозування і пошук закономірностей і є інструментом аналітика, менеджера або особи, відповідальної за прийняття рішень і формування політики компанії.

Спочатку дані знаходяться в реляційній базі даних, куди вони потрапляють завдяки програмі, що займається обліком даних. Це може бути система складського обліку, реєстрації замовлень клієнтів, укладання ордерів і т. ін. Ця база даних у більшості випадків має OLTP (On-Line Transactional Processing) структуру, що називається так через те, що вона оптимізована для обслуговування коротких поновлювальних транзакцій, кожна з яких стосується порівняно невеликої кількості таблиць

бази даних [3], [5]. Максимально нормалізована структура бази відповідає саме задачам обліку та реєстрації інформації.

У процесі аналізу облікова інформація подається та досліджується з різних позицій, у різних аспектах. При цьому формуються аналітичні запити, приміром такі:

- У якому регіоні був досягнутий максимальний рівень збуту даної продукції торік?

- У якої категорії покупців вона користувалася найвищим попитом?

- Якою була динаміка продажів по цьому регіону в детальнішому часовому масштабі?

- Який прогноз попиту дає тренд на II квартал цього року? та інше.

Структуру транзакційной бази не оптимізовано для таких запитів, оскільки вони задіюють усі або майже усі таблиці в базі (а відпрацьовування численних зв'язків між ними — витратна щодо часу та ресурсів операція) і, подовгу блокуючи дані на читання, перешкоджають оперативному проходженню OLTP-транзакцій у системі обліку. Отже, для опрацювання OLAP-запитів необхідний окремий формат збереження даних.

1. Основні поняття OLAP

У процесі аналізу кожний аналізований факт зручно розглядати як функцію від його характеристик. Наприклад, продаж є функцією від товару, покупця, продавця, місця і часу здійснення угоди, можливо, ще якихось істотних параметрів. Сукупність такої інформації може бути раціонально подана у

вигляді інтуїтивно зрозумілої моделі даних — багатовимірного куба (Cube). Осями багатовимірної системи координат слугують основні атрибути аналізованого бізнес-процесу [3], [5]. Так, для продажів це можуть бути товар, регіон, тип покупця. Як один із вимірів може використовуватися час. На перетинаннях осей — вимірів (Dimensions) — знаходяться дані, що кількісно характеризують процес, — міри (Measures). Це можуть бути обсяги продажів у штуках або у грошовому вимірі, залишки на складі, витрати і т. ін. Користувач, що аналізує інформацію, може «розрізати» куб за різними напрямками, одержувати зведені (наприклад, по роках) або, навпаки, детальні (по тижнях) відомості, здійснювати інші маніпуляції, що будуть потрібні в процесі аналізу. У тривимірному кубі, зображеному на рис. 1, мірами є суми продажів, а вимірами — час, товар і регіон.

Значення, що «відкладаються» вздовж вимірів, називаються членами виміру (members). Члени виміру використовуються як для «розрізування» куба, так і для відбору (фільтрації) даних — коли аналітика цікавлять у вимірі не всі значення, а їх підмножина (наприклад, три міста з декількох десятків). Значення членів виміру відображаються у двовимірному представленні куба, як заголовки рядків і стовпців.

Члени виміру можуть утворювати ієрархії, що складаються з одного або декількох рівнів. Наприклад, для виміру «Географія» типовою ієрархією може бути «Світ → континенти → країни → регіони → міста». На кожному рівні

може знаходитися декілька членів, наприклад, «Міста» = {Київ, Полтава, Дніпропетровськ, Харків}. В однім вимірі можна реалізувати більше однієї ієрархії. Наприклад, для часу: {Рік, Квартал, Місяць, День} і {Рік, Тиждень, День}.

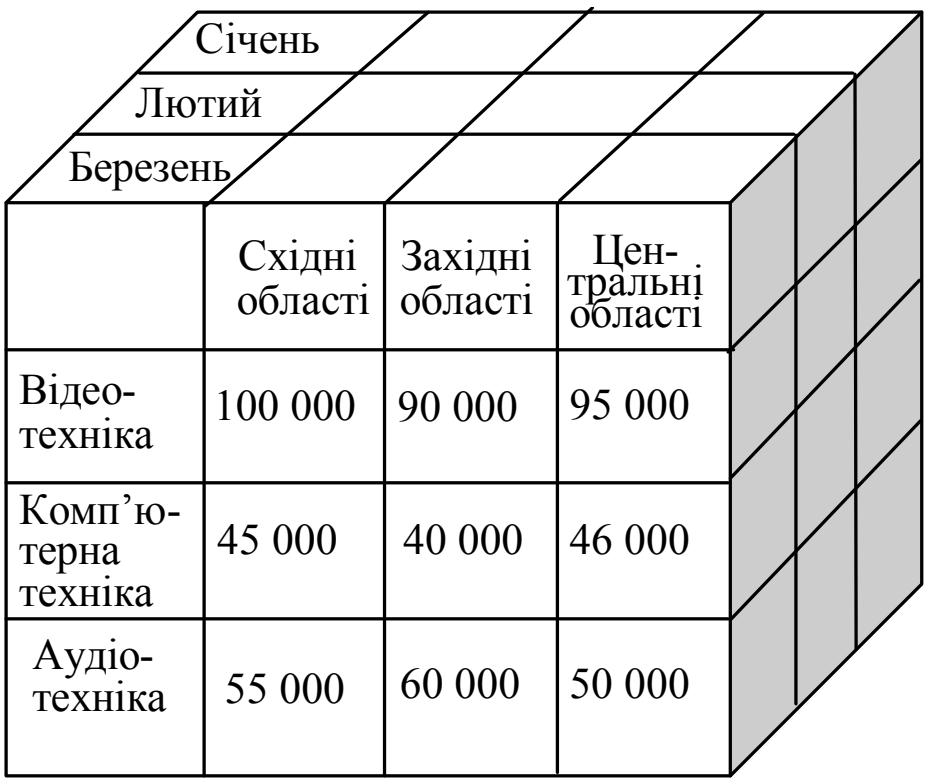


Рисунок 1 - Приклад куба

Для візуалізації даних, що зберігаються в багатовимірному кубі, застосовуються, як правило, звичні двовимірні, тобто табличні, представлення, що мають складні ієрархічні заголовки рядків і стовпців. Двовимірне представлення куба можна одержати, «розрізавши» його поперек однієї або декількох осей (вимірів). При цьому фіксують значення усіх, крім двох, вимірів і одержують звичайну двовимірну таблицю. У горизонтальній осі таблиці (заголовки стовпців) представлений один вимір, у вертикальній (заголовки рядків) — інший, а у комірках таблиці — значення

мір. За необхідності показу декількох мір, набір мір може розглядатися, як один з вимірів — тоді одну з осей таблиці займуть назви мір, а іншу — значення єдиного «нерозрізаного» виміру.

Основними операціями над вимірами є заглиблення (drilldown — перехід усередину по ієрархії на детальніші рівні) і скручування (rollup — навпаки, прямування вгору по ієрархії для одержання масштабнішої картини). Наприклад, продажі по Сходу України складаються з продажів по Харківській, Донецькій і Луганській областях. Для мір здебільшого використовуються агрегуючі операції (sum, count і т. ін.), за допомогою яких складаються значення мір для членів вищих рівнів ієрархії.

З метою прискорення опрацювання аналітичних запитів ще на стадії наповнення сховища даних, прагнуть підрахувати агрегати для членів вищих рівнів ієрархій (наприклад, по регіону) на основі детальних значень, що містяться в реляційній базі обліку. Отримавши запит на продажі по деякому регіону, процесор OLAP візьме вже готову, завчасно розраховану величину, домігшись тим самим мінімального часу опрацювання запиту.

При занесенні даних у сховище служби перетворення даних здійснюють, крім попередньої агрегації, доступ до різнорідних даних, очищення їх, перевірку на несуперечливість та уніфікацію.

Абревіатуру OLAP (On-Line Analytical Processing) уперше ввів відомий вчений у галузі баз даних, творець широко

розповсюдженої реляційної моделі Кодд (E. F. Kodd) у праці «Providing OLAP to User Analysis: An IT Mandate» (1993 р.), де він сформулював 12 основних правил, яким мають задовольняти OLAP-системи. У 1995 р. до цих правил було додано ще шість. Кодд розбив правила, назвавши їх «особливостями», на такі чотири групи: основні особливості, спеціальні особливості, особливості представлення звітів та управління вимірами.

1. Основні особливості.

1.1. Багатовимірне концептуальне представлення даних (оригінальне правило 1). Ця особливість являє собою головну характеристику OLAP.

1.2. Інтуїтивне маніпулювання даними (оригінальне правило 10). На думку Е. Ф. Кодда, маніпулювання даними має здійснюватися за допомогою прямих дій над комірками в режимі перегляду без використання меню і множинних операцій.

1.3. Доступність: OLAP як посередника (оригінальне правило 3). У цьому правилі підкреслюється роль OLAP, як прошарку між гетерогенними джерелами даних і представленням для кінцевого користувача.

1.4. Пакетне здобування проти інтерпретації (нове правило). Це правило потребує, щоб продукт в однаковій мірі ефективно забезпечував доступ як до власного сховища даних, так і до зовнішніх даних. По суті, Е. Ф. Кодд наполягав на багатовимірному представленні даних із частковими попередніми обчисленнями для великих багатовимірних баз

даних, щоб будь-які детальні дані були прозорі та доступні. Сьогодні це відповідає визначенню гібридних OLAP, що стають найбільш популярною архітектурою.

1.5. Моделі аналізу OLAP (нове правило). OLAP-продукти мають підтримувати чотири моделі аналізу, описані Коддом у статті: категоріальний, тлумачний, умоглядний і стереотипний (формування параметрично налаштовуваних звітів, формування розрізів та угруповань з обертанням, аналіз у стилі «що-якщо» і моделі пошуку цілей).

1.6. Архітектура «клієнт-сервер» (оригінальне правило 5). OLAP-продукт має бути не тільки клієнт-серверним, а й серверний компонент має бути досить інтелектуальним для того, щоб різноманітні клієнти могли підключатися з мінімумом зусиль і програмування.

1.7. Прозорість (оригінальне правило 2). Повна відповідність цьому правилу означає, що, наприклад, користувач електронної таблиці спроможний одержати всі необхідні дані з OLAP-машини, не підозрюючи, звідки вони в кінцевому рахунку беруться. Щоб виконати це, продукт має забезпечувати безпосередній доступ до гетерогенних джерел даних і одночасно мати вбудовану повнофункціональну електронну таблицю. Більшість продуктів не надають або доступу до електронних таблиць, або безпосереднього доступу до гетерогенних джерел.

1.8. Багатокористувацька підтримка (оригінальне правило 8). OLAP-програми не повинні працювати лише в режимі читання даних. Інструменти OLAP мають

забезпечувати одночасний доступ (читання і запис), інтеграцію і конфіденційність. Це правило вказує на стратегічний напрям розвитку OLAP.

2. Спеціальні особливості.

2.1. Обробка ненормалізованих даних (нове правило). Воно вказує на необхідність інтеграції між OLAP-машиною і ненормалізованими джерелами даних. Модифікації даних, виконані в середовищі OLAP, не повинні призводити до змін даних, збережених у вихідних зовнішніх системах.

2.2. Зберігання результатів OLAP: збереження їх окремо від вихідних (початкових) даних (нове правило). OLAP-програми, що працюють у режимі читання-запису, не повинні впливати безпосередньо на оброблювані дані, і дані, модифіковані в OLAP, мають зберігатися окремо від даних транзакцій.

2.3. Виключення відсутніх значень (нове правило). Відсутні значення повинні відрізнятися від нульових значень. Це істотно з погляду компактності збереження даних

2.4. Обробка відсутніх значень (нове правило). Всі відсутні значення будуть ігноруватися OLAP-аналізатором.

3. Особливості представлення звітів.

3.1. Гнучкість формування звітів (оригінальне правило 11). Виміри мають бути розміщені в звіті так, як це потрібно користувачеві. Бажано, щоб засоби аналізу та можливості формування звітів були комбіновані в одному модулі.

3.2. Стандартна продуктивність звітів (оригінальне правило 4). Продуктивність формування звітів не повинна

істотно падати із збільшенням кількості вимірів і розмірів бази даних.

3.3. Автоматичне настроювання фізичного рівня (заміна оригінального правила 7). OLAP-системи повинні автоматично настроювати свою фізичну схему залежно від типу моделі, об’ємів даних і розрідженості бази даних.

4. Управління вимірами.

4.1. Універсальність вимірів (оригінальне правило 6). Усі виміри мають бути рівноправні у структурі та операційних можливостях.

4.2. Необмежене число вимірів і рівнів агрегації (оригінальне правило 12).

4.3. Необмежені операції між розмірностями (оригінальне правило 9). Цей тип операцій є важливим, якщо користувачеві потрібні комплексні складні дослідження, а не тільки перехресні вибірки даних, зокрема, вони доречні в програмах аналізу рентабельності.

У 1995 р. Пендс (Nigel Pendse) та Кріт (Richard Creeth) переробили ці вимоги і запропонували власне визначення характеристик OLAP, звівши його до так званого тесту FASMI (Fast Analysis of Shared Multidimensional Information), що означає «швидкий аналіз розділюваної багатовимірної інформації».

FAST (швидкий) — означає, що система повинна забезпечувати видачу більшості відповідей користувачам у межах приблизно п’ятьох секунд. При цьому найпростіші запити опрацьовуються протягом однієї секунди і дуже

небагато — понад 20 секунд. Постачальники вдаються до широкого розмаїття методів, щоб досягти цієї мети, включаючи спеціалізовані форми збереження даних, великі попередні обчислення або ж підсилюючи апаратні вимоги.

ANALYSIS (аналіз) — означає, що система може справлятися з будь-яким логічним і статистичним аналізом, характерним для даної програми, і надає його можливості у вигляді, доступному для кінцевого користувача. Необхідно дозволити користувачу визначати нові спеціальні обчислення як частину аналізу і формувати звіти будь-яким бажаним засобом, без необхідності програмування. Засоби аналізу могли б включати певні процедури типу аналізу часових рядів, розподілу витрат, валютних переведень, пошуку цілей, зміни багатовимірних структур, непроцедурного моделювання, виявлення виняткових ситуацій, витягів даних та інші операції, залежні від програми.

SHARED (розділювача) означає, що система здійснює усі вимоги захисту конфіденційності (можливо до рівня комірки) і, якщо множинний доступ для запису необхідний, забезпечує блокування модифікацій на відповідному рівні. Не в усіх програмах існує необхідність зворотного запису даних. Проте кількість таких програм збільшується, і система повинна бути спроможна опрацювати множинні модифікації безпечним засобом. Це — головне дошкульне місце багатьох OLAP-продуктів, які мають тенденцію припускати, що в усіх програмах OLAP потрібне тільки читання, і надають спрощені засоби захисту.

MULTIDIMENSIONAL (багатовимірна) — система повинна забезпечити багатовимірне концептуальне представлення даних, включаючи повну підтримку для ієрархій і множинних ієрархій, оскільки це найбільш логічний спосіб аналізувати бізнес організації.

INFORMATION (інформація) — необхідна інформація має бути отримана там, де в ній є потреба. Потужність різноманітних продуктів може бути вимірювана тим, скільки вхідних даних вони можуть опрацьовувати, а не скільки гігабайт вони можуть зберігати. Потужність продуктів дуже різноманітна — найбільші OLAP-продукти можуть оперувати, принаймні, в тисячу разів більшою кількістю даних у порівнянні із найменшими. При цьому варто враховувати багато факторів, включаючи дублювання даних, необхідну оперативну пам'ять, використання дискового простору, експлуатаційні показники, інтеграцію з інформаційними сховищами і т. ін.

2. Архітектура OLAP-програм

OLAP-програми можуть мати три архітектурні рівні [3], [5]:

- Багатовимірне подання даних — засоби кінцевого користувача, що забезпечують багатовимірну візуалізацію і маніпулювання даними; прошарок багатовимірного подання абстрагований від фізичної структури даних і сприймає дані як багатовимірні.

- Багатовимірна обробка — засіб (мова) формулювання багатовимірних запитів (традиційна реляційна мова SQL тут виявляється непридатною) і процесор, що може опрацювати і виконати такий запит.

- Багатовимірне збереження — засоби фізичної організації даних, що забезпечують ефективне виконання багатовимірних запитів.

Перші два рівні в обов'язковому порядку присутні в усіх OLAP-засобах. Третій рівень, хоча і є широко розповсюдженим, не є обов'язковим, оскільки дані для багатовимірного представлення можуть витягатися і зі звичайних реляційних структур; процесор багатовимірних запитів у цьому випадку транслює багатовимірні запити в SQL-запити, що виконуються реляційною СУБД.

Конкретні OLAP-продукти здебільшого являють собою або засіб багатовимірного представлення даних, OLAP-клієнт (наприклад, Pivot Tables в Excel 2000 фірми Microsoft або ProClarity фірми Knosys), або багатовимірну серверну СУБД, OLAP-сервер (наприклад, Oracle Express Server або Microsoft OLAP Services). Прошарок багатовимірної обробки звичайно буває вбудований в OLAP-клієнт і/або в OLAP-сервер, але може бути виділений у чистому вигляді, як, наприклад, компонент Pivot Table Service фірми Microsoft.

OLAP-сервери, або сервери багатовимірних БД, можуть зберігати свої багатовимірні дані по-різному. Як детальні дані, так і агрегати можуть зберігатися або в реляційних, або в багатовимірних структурах. Багатовимірне збереження дає

змогу поводитися з даними як із багатовимірним масивом, завдяки чому забезпечуються однаково швидкі обчислення сумарних показників і різноманітні багатовимірні перетворення за будь-яким з вимірів. Нещодавно OLAP-продукти підтримували або реляційне, або багатовимірне збереження. Зараз, здебільшого, той самий продукт забезпечує обидва ці види збереження, а також третій вид — змішаний. Застосовуються такі терміни:

MOLAP (Multidimensional OLAP) — і детальні дані, і агрегати зберігаються у багатовимірній БД;

ROLAP (Relational OLAP) — детальні дані залишаються там, де вони знаходилися початково, — у реляційній БД; агрегати зберігаються у тій же БД у спеціально створених службових таблицях;

HOLAP (Hybrid OLAP) — детальні дані зберігаються у реляційній БД, а агрегати — у багатовимірній БД.

Кожний із цих способів має свої переваги і хиби і повинен застосовуватися залежно від умов — обсягу даних, потужності реляційної СУБД тощо.

3. Засоби OLAP-аналізу компанії Microsoft

У комплект Microsoft SQL Server 7.0 входить повнофункціональний OLAP-сервер — OLAP Services for SQL Server. Для обслуговування запитів клієнтів сервер використовує спеціальний протокол взаємодії і мову запитів. Наприклад, для взаємодії клієнта із серверною реляційною СУБД — SQL Server — використовуються протоколи ODBC

або OLE DB і мова запитів SQL. Для доступу до OLAP-серверу компанією Microsoft були розроблені протокол OLE DB for OLAP і мова запитів до багатовимірних даних — MDX (MultiDimensional eXpression). Аналогічно тому, як для спрощення і зручності над OLE DB було розроблено прошарок об'єктів ADO (Active Data Objects), над OLE DB for OLAP побудовано ADO MD (Multidimensional ADO). У ролі OLAP-клієнта може використовуватися Microsoft Excel 2000.

OLAP-сервер — OLAP Services — функціонує як сервіс Windows NT. Для його адміністрування використовується набір об'єктів Decision Support Objects (DSO) — аналог Distributed Management Objects (DMO) для SQL Server. Основний засіб адміністрування OLAP Services — OLAP Manager — побудовано на базі DSO. Він є модулем розширення (snap-in) для MMC (Microsoft Management Console).

За допомогою OLAP Manager адміністратор створює бази даних і будує куби. У цьому процесі йому може допомогти майстер Cube Wizard, що здійснює формування кубів із реляційних баз даних, побудованих за схемою «зірки» або «сніжинки». Ці схеми, традиційні для сховищ даних, складаються з центральної таблиці, називаної таблицею фактів, і декількох пов'язаних із нею таблиць атрибутів (вимірів). При створенні кубів таблиця фактів (fact table) перетвориться на набір мір (measures), а таблиці атрибутів — на виміри (dimensions). Адміністратор указує таблиці фактів та атрибутів, а також вибирає, які саме поля використовуватимуться як міри і виміри.

Вибір технології збереження здійснюється за допомогою майстра Storage Designe Wizard. Для кожного куба адміністратор може вибрати будь-яку з трьох технологій збереження — MOLAP, ROLAP або HOLAP. MOLAP рекомендується використовувати в тому разі, коли потрібний швидкий доступ до невеликого або середнього обсягу даних. Для роботи з великими обсягами даних, особливо у разі невисокої інтенсивності запитів до них (наприклад, при роботі з архівними даними), рекомендується використовувати ROLAP. HOLAP сполучає у собі переваги перших двох технологій. При використанні будь-якої з цих технологій Microsoft SQL Server OLAP Services не зберігає порожніх значень, вирішуючи в такий спосіб проблему розріджених даних.

Побудова агрегатів прискорює виконання OLAP-запитів, але призводить до «розбухання» даних. Ранні OLAP-продукти потребували обчислення всіх агрегатів, що були так чи інакше необхідні для аналізу. Природно, це викликало лавиноподібне зростання обсягів. OLAP Services дає змогу вирішити цю проблему завдяки «інтелектуальному» агрегуванню даних. При створенні куба майстер Data Storage and Aggregation Wizard, застосовуючи евристичні методики, визначає, які агрегати потрібно обчислити, щоб досягти заданої продуктивності для більшості можливих запитів при мінімальному обсязі. При цьому використовується відоме правило 20/80, характерне для степеневих залежностей, відповідно до якого близько 80 % від максимально можливого числа агрегатів не здійснюють істотного впливу на зростання обсягу сховища. Стадія

вибухового зростання, як правило, починається на останніх 20 % агрегатів. Евристичний алгоритм у OLAP Services аналізує модель метаданих сховища і визначає оптимальну множину агрегатів, обчислення яких дасть максимальний виграш у продуктивності. Ця множина являє собою базовий набір агрегатів, від яких найлегше можуть бути розраховані інші агрегати, що не обчислюються на етапі наповнення сховища. Коли в процесі обробки запиту потрібно одержати значення похідного агрегату, який завчасно не розраховувався, він розраховується не шляхом сканування детальних даних, а на основі базових попередньо розрахованих агрегатів.

Іншою можливістю зменшення обсягу при обчисленні агрегатів є обмеження обсягу даних, який призначається для збереження агрегатів, після чого майстер вибирає найефективнішу схему побудови їх. При цьому він враховує можливість побудови одних агрегатів з інших — наприклад, маючи дані про сумарні продажі за квітень, травень і червень, можна легко обчислити обсяг продажів за II квартал.

Реальні запити можуть відрізнятися від тих, що розглядалися при початковій побудові агрегатів. Для того щоб оптимізувати продуктивність виходячи з реального використання, OLAP Services може записувати запити користувачів у спеціальний журнал. Потім цей журнал використовується майстром Usage Based Optimization Wizard для створення нової, досконалішої схеми агрегування. При цьому Usage-Based Optimization Wizard читає журнал транзакцій за цей період часу і визначає агрегати, обчислення

яких могло б прискорити виконання цих запитів. Адміністратор може обмежити список запитів у журналі, наприклад, указавши майстру розглядати тільки ті з них, час відповіді на які зайняв понад n сек.

У рамках одного логічного куба адміністратор може організувати декілька фізичних розділів (partitions). Розділи можуть мати різноманітну структуру збереження (MOLAP, ROLAP, HOLAP) і різний рівень агрегованості. У варіанті SQL Server Enterprise Edition розділи можуть фізично розташовуватися на різних серверах. При організації розділів адміністратор повинен стежити, щоб дані у розділах не перетиналися й водночас складали повну картину, інакше звіти даватимуть неправильні результати. Розділи можуть спиратися на різні таблиці фактів у вихідній реляційній БД або використовувати одну таблицю фактів, розділену на частини умовами фільтрації.

Мета використання розділів — оптимізація швидкості доступу до даних залежно від частоти використання їх. Для часто використовуваних даних поточного року адміністратор може задати збереження типу MOLAP і високий ступінь агрегованості, а для архівних даних — ROLAP і мінімум агрегатів. Наприклад, куб, що містить дані про продажі, може бути розбитий на декілька фрагментів (partitions), один з яких із даними за поточний рік запитується досить часто, а інші — рідше. Тоді, створюючи фрагмент як перетин по поточному року, вибирають формат MOLAP, оскільки тут найбільш критичним питанням виступає продуктивність, для перетину по

минулому року — HOLAP, а для інших історичних даних — ROLAP. При використанні Enterprise Edition розділи дають змогу організувати ефективний розподіл навантаження між серверами. За будь-якої організації розділів їх структура не видна кінцевому користувачу — він бачить логічно єдиний куб.

За будь-якої технології організації куба дані в ньому потрібно періодично обновляти. При використанні ROLAP або HOLAP потрібно тільки перераховувати агрегати (тому що детальні дані обновлюються природним шляхом при поповненні сховища), а при використанні MOLAP необхідно також оновити детальні дані. Відновлення даних може здійснюватися в один із трьох способів:

Process — це повне відновлення як структури куба, так і всіх його даних. Воно застосовується при початковому наповненні куба або у разі зміни структури агрегатів;

Incremental update — додавання нових даних із відновленням агрегатів. Інкрементне відновлення являє собою найшвидший варіант, який додає в куб тільки нові дані, що з’явилися в реляційній базі з моменту останньої повної обробки або відновлення;

Refresh Data — повне відновлення даних та агрегатів без зміни структури куба. Воно застосовується в тому разі, коли в реляційній базі, інформаційно пов’язаній зі сховищем, оновилися дані, які вже належать сховищу.

Служба OLAP Services дає змогу управляти правами доступу користувачів до кубів. Користувачі відповідно до їх імен ідентифікуються в Windows NT. Права видаються на куб

цілком і можуть бути трьох видів: «читання», «читання-запис» та «адміністрування». Управління правами можливе тільки в тому разі, якщо куби зберігаються на диску із файловою системою NTFS.

Служба OLAP Services забезпечує також низку розширених можливостей OLAP-аналізу. До них належать:

- віртуальні куби. Вони подібні представленням (View) у реляційних базах і дають змогу організувати нове представлення даних, логічно пов’язуючи декілька кубів за спільними вимірами. Віртуальні куби не займають додаткового дискового простору;

- властивості вимірів (member properties). Дають змогу описати додаткові атрибути для кожного значення деякого виміру. Наприклад, для кожного магазину можна ввести атрибут, що зберігає прізвище менеджера, який управляє цим магазином. Це дає змогу внести додатковий атрибут у дані й надалі візуалізувати його і використовувати при доборі даних;

- віртуальні виміри. Властивості вимірів також дають змогу організувати віртуальні виміри, що подібні до звичайних вимірів, але не потребують місця для збереження. Агрегати по віртуальних вимірах не зберігаються і завжди обчислюються оперативно;

- запити «що — якщо» і модифікація куба (write back). Дані в куб потрапляють із сховища і надалі не змінюються, тому що це, по суті, історичні дані. Проте користувачі OLAP Services мають можливість приписувати зміни до даних у кубі. Можливість клієнтських застосувань оновлювати дані в

комірках куба пов’язана з підтримкою запитів типу «що — якщо» і зворотного запису (writeback) у куб. Обидві ці риси посилюють можливості OLAP Services як інструмента бізнес-аналізу, надаючи аналітику можливість одержувати відповіді на умовні запити (наприклад, «Що сталося б з продажами в Київській області за попередній місяць, якби ціни на продукт А були збільшені на 10 %, а транспортні витрати знижені в 1,3 рази?»). Запис значень у комірки здійснюється через інтерфейс OLE DB for OLAP і має бути захищений контекстом транзакції. До завершення транзакції всі зміни поміщаються у кеш на клієнті і стають одразу помітними тільки в рамках поточної OLE DB-сесії. Випадок «відката» (undo) транзакції відповідає ситуації «що — якщо», тобто зміні вхідних даних, перегляду результатів і поверненню до початкового стану. Фіксація транзакції відповідає зворотному запису. При цьому зміни можуть бути внесені тільки в куб, позначений як READ-WRITE, і тільки користувачем, що володіє правами на відновлення куба. Після фіксації (Commit) транзакції решта сесій одержують можливість побачити внесені зміни;

Функції, визначені користувачем. OLAP Services підтримує створення і використання визначених користувачем функцій (UDF). Функції приймають аргументи і повертають значення в термінах синтаксису MDX. Для створення їх може залучатися будь-який засіб розроблення, спроможний генерувати бібліотеки ActiveX-об’єктів (Visual Basic, Visual C++ і т. ін.).

Служба Pivot Tables Services (PTS) виступає як постачальник OLE DB for OLAP, що забезпечує виконання

запиту будь-якого клієнта OLE DB for OLAP. Кожний клієнт підключається до OLAP Services через Pivot Table Services, що виступає в ролі драйверу, який здійснює диспетчеризацію з'єднання. При цьому PTS містить ряд розширень щодо стандарту OLE DB, таких як кешування запитів на клієнті та робота з локальними кубами. Названі його властивості забезпечують такі можливості для клієнтських OLAP-інструментів.

Кешування на клієнті дає змогу мінімізувати звертання до серверу. Наприклад, збережені в локальному кеші цифри квітня, травня і червня дають змогу обчислити показники кварталу без звертання до серверу. PTS «розуміє» структуру серверних кубів, тому що при першому звертанні до OLAP-серверу зчитує метадані, що описують цю структуру.

Найважливішою можливістю PTS є зберігання результатів запитів до OLAP-серверу у вигляді локальних кубів, що потім можна аналізувати, не звертаючись до серверу. PTS містить частину програмного коду OLAP Services, завдяки чому забезпечує виконання OLAP-запитів як до локальних кубів, так і до реляційних джерел даних. Фактично PTS є настільним OLAP-сервером, позбавленим, проте, можливості зберігати агрегати. Тому його швидкодія зворотно пропорційна обсягу даних.

За допомогою PTS локальний куб може бути створений OLAP-клієнтом не тільки в результаті запиту до OLAP-серверу, а й на основі запиту до реляційних даних, як до серверних, так і до локальних. Крім того, PTS дає змогу вбудовувати засоби

OLAP-аналізу в Web-сторінки, що звертаються до PTS через набір об’єктів ADO MD.

4. Засоби аналізу даних у Microsoft Office 2000

Microsoft Excel 2000 містить новий механізм зведених таблиць — OLAP Pivot Tables, що замінив собою однойменний механізм попередніх версій. Поряд із попередніми можливостями аналізу реляційних даних механізм Pivot Tables тепер включає можливості аналізу OLAP-даних, тобто виступає як OLAP-клієнт. Як сервер може використовуватися Microsoft SQL Server 7.0, а також будь-який продукт, що підтримує інтерфейс OLE DB for OLAP. Механізм зведених таблиць Excel у повному обсязі підтримує можливості, що надаються описаним вище сервісом Pivot Table Services (PTS). Таким чином, аналізовані OLAP-дані можуть знаходитися як у локальних кубах, так і на OLAP-сервері.

Використання можливостей OLAP дає користувачам Excel такі переваги:

- значно підвищується швидкість обробки аналітичних запитів;

- модель даних OLAP проста й інтуїтивно зрозуміла користувачеві;

- механізм зведених таблиць в Excel 97 потребував завантаження всіх аналізованих реляційних даних у кеш, що організовувався в оперативній пам’яті клієнтського комп’ютера. Це накладало жорсткі обмеження як на об’єм, так і на швидкість обробки. Нові зведені таблиці можуть працювати

в режимі «клієнт-сервер» і при використанні OLAP-серверу одержують з нього тільки необхідні для показу дані. Найчастіше необхідні зведені значення обчислено на сервері заздалегідь, що різко підвищує швидкість виконання запитів.

Використання OLAP-даних у зведених таблицях Excel 2000 (у порівнянні з використанням реляційних даних) певною мірою є менш гнучким. Те, що в кубі описане як вимір, не може стати мірою, і навпаки. Не можна вводити довільні рівні підсумовування — використовуються лише ті, що визначені в кубі. Якщо користувачам потрібні нові рівні підсумовування або інші зміни структури куба, то вони повинні звернутися до адміністратора OLAP-серверу. У наступних випусках Excel компанія Microsoft планує зняти ці обмеження.

Microsoft Office 2000 містить також набір ActiveX-компонентів, називаних Office 2000 Web Components, що дають змогу організувати аналіз OLAP-даних засобами перегляду Web. До них належать такі чотири компоненти:

- Spreadsheet — реалізує обмежену функціональність аркуша Excel;

- PivotTable — аналог зведених таблиць Excel; може працювати з даними OLAP Services;

- Chart — дає змогу будувати діаграми, що базуються як на реляційних, так і на OLAP-даних;

- Data Source — службовий компонент для прив’язки інших компонентів до джерела даних.

При роботі з OLAP-даними Web Components звертаються до Pivot Table Services.

Термінологічний словник

Виміри (Dimensions) - осі багатовимірного гіперкуба, у ролі яких виступають основні атрибути аналізованого бізнес-процесу.

Гіперкуб (куб, Cube) - багатовимірна модель даних, що є інтуїтивно зрозумілою і використовується для аналізу ділової інформації.

Життєвий цикл ПЗ - модель створення і супроводження ПЗ, що відображає його різні стани, починаючи з моменту виникнення ідеї створення ПЗ і закінчуючи моментом виходу його з експлуатації.

Міри (Measures) - дані, що кількісно характеризують процес і знаходяться всередині гіперкуба на перетинаннях гіперплощин, що відповідають міткам.

Мітки (члени виміру, Members) - значення, що відкладаються вздовж виміру. Мітки використовуються як для «розрізування» куба, так і для обмеження (фільтрації) даних. Мітки можуть утворювати ієрархії.

Репозиторій (repository) - база даних CASE, в якій зберігається вся проектна інформація

Сховище даних (Data Warehouse) - предметно-орієнтоване, прив’язане до часу і незмінне зібрання даних для підтримки процесу управлінських рішень. Дані у сховище потрапляють з оперативних систем (OLTP-систем), призначених для автоматизації бізнес-процесів. Крім того, сховище даних може поповнюватись із зовнішніх джерел, наприклад, статистичних звітів

CASE (Computer-Aided Software/System Engineering) - сукупність методологій аналізу, проектування, розроблення і супроводження складних систем програмного забезпечення, підтримана комплексом взаємозв'язаних засобів автоматизації.

DFD (Data Flow Diagram) - діаграма потоків даних. На DFD відображаються потоки даних, процеси перетворення вхідних потоків на вихідні, сховища інформації, джерела і споживачі інформації, зовнішні щодо системи. Надалі ці діаграми є підґрунтям для формування структури розроблюваного ПЗ.

ERD (Entity-Relationship Diagram) - діаграма «сутність-зв'язок». На ERD показують так звані сутності (інформаційні об'єкти, що становлять інтерес з погляду наступного зберігання) і зв'язки між сутностями з відображенням характеру зв'язку. Ці діаграми є основою для проектування баз даних.

SC (Structure Chart) - структурна карта. SC слугують для відображення архітектури системи у вигляді сукупності програмних модулів і зв'язків між ними, а також даних, що передаються від одного модуля до іншого.

STD (State Transition Diagram) - діаграма переходів станів. На STD відображуються стани, в яких може перебувати система, і можливі переходи з одного стану до іншого. Згідно з діаграмою проектується інтерфейс користувача.

FAST (швидкий) - означає, що система повинна забезпечувати видачу більшості відповідей користувачам у межах приблизно п'ятьох секунд. При цьому найпростіші

запити опрацьовуються протягом однієї секунди і дуже небагато - понад 20 секунд. Постачальники вдаються до широкого розмаїття методів, щоб досягти цієї мети, включаючи спеціалізовані форми збереження даних, великі попередні обчислення або ж підсилюючи апаратні вимоги.

ANALYSIS (аналіз) - означає, що система може справлятися з будь-яким логічним і статистичним аналізом, характерним для даної програми, і надає його можливості у вигляді, доступному для кінцевого користувача. Необхідно дозволити користувачу визначати нові спеціальні обчислення як частину аналізу і формувати звіти будь-яким бажаним засобом, без необхідності програмування. Засоби аналізу могли б включати певні процедури типу аналізу часових рядів, розподілу витрат, валютних переведень, пошуку цілей, зміни багатовимірних структур, непроцедурного моделювання, виявлення виняткових ситуацій, витягів даних та інші операції, залежні від програми.

SHARED (розділювача) - означає, що система здійснює усі вимоги захисту конфіденційності (можливо до рівня комірки) і, якщо множинний доступ для запису необхідний, забезпечує блокування модифікацій на відповідному рівні. Не в усіх програмах існує необхідність зворотного запису даних. Проте кількість таких програм збільшується, і система повинна бути спроможна опрацювати множинні модифікації безпечним засобом. Це - головне дошкульне місце багатьох OLAP-продуктів, які мають тенденцію припускати, що в усіх

програмах OLAP потрібне тільки читання, і надають спрощені засоби захисту.

MULTIDIMENSIONAL (багатовимірна) - система повинна забезпечити багатовимірне концептуальне представлення даних, включаючи повну підтримку для ієрархій і множинних ієрархій, оскільки це найбільш логічний спосіб аналізувати бізнес організації.

INFORMATION (інформація) - необхідна інформація має бути отримана там, де в ній є потреба. Потужність різноманітних продуктів може бути вимірювана тим, скільки вхідних даних вони можуть опрацьовувати, а не скільки гігабайт вони можуть зберігати. Потужність продуктів дуже різноманітна - найбільші OLAP-продукти можуть оперувати, принаймні, в тисячу разів більшою кількістю даних у порівнянні із найменшими. При цьому варто враховувати багато факторів, включаючи дублювання даних, необхідну оперативну пам'ять, використання дискового простору, експлуатаційні показники, інтеграцію з інформаційними сховищами і т. ін

OLAP (OnLine Analytical Processing) - оперативний аналіз даних. Сукупність засобів багатовимірного аналізу даних, накопичених у сховищі.

Computer Aided of Logistics Support - автоматизовані логістичні системи.

Computer Aided Acquisition and Support - автоматизовані постачання і підтримка.

Computer Aided Acquisition and Life-Cycle Support - автоматизація безперервних постачань і життєвого циклу.

Commerce At Life Speed - бізнес у високому темпі.

Information Engineering - інформаційна інженерія.

Artificial Intelligence - штучний інтелект.

Re - engineering - воротне проектування.

Business Process Reengineering - реінжиніринг бізнес-процесів.

Multi - Agent Systems - багатоагентні системи.

Knowledge Management - управління знаннями.

Industrial Engineering - промислова інженерія.

Total Quality Management - управління якістю.

Computer Aided Software Engineering - CASE –технології.

Just - in - time - точно вчасно.

OPT (Optimised Production Technology - оптимізаційна технологія виробництва.

CIM (Computer Integrated Manufacturing - інтегровані виробництва на основі обчислювальної техніки.

CALS (Continuous Acquisition and Life Circle Support - підтримка безперервного життєвого циклу продукції).

ERP, MRP (Material Requirements Planning - планування потреб в матеріалах.

MRP II (Manufacturing Recourse Planning - планування ресурсів виробництва.

Total Quality Management - технологія розробки систем якості.

Information and Communication Technologies - інформаційні технології.

Перелік скорочень, символів, спеціальних термінів

АРВЗ – алгоритм рішення винахідницьких задач

АРМ – автоматизоване робоче місце

АСНД – автоматизована система наукових досліджень

САПР – система автоматизованого проектування

АСКТП – автоматизована система керуванням технологічними процесами

АСТПВ – автоматизована система технологічної підготовки виробництва

АСУП – автоматизована система управління виробництвом на рівні підприємства

АСУ – автоматизована система управління виробництвом на рівні галузі

БД – база даних

БНД – банк даних

ВЕТ – виріб електронної техніки

ВІС – велика інтегральна схема

ВЗП – зовнішній запам'ятовуючий пристрій

ВР – варіант рішення

ГСАПР – гнучка САПР

ДМП – дискретний монтажний простір

ДС – діалогова система

ДСТУ – державний стандарт України

ЕОМ – електронна обчислювальна машина

ЕРЕ – електрорадіоелемент

ЄСКД – єдина система конструкторської документації

ЖЦ – життєвий цикл

ЗВІС – зверхвелика інтегральна схема

ЗОТ – засіб обчислювальної техніки

ІГС – інтерактивна-графічна станція

ІЗ – інформаційне забезпечення

ІНСАПР – інтегрована САПР

ІС – інтегральна схема

ІТ – інформаційні технології

ІТП – інженерно-технічний персонал

КГА – креслярсько - графічний автомат

КД – конструктивні дані

КРЕМКОМ – кремнієвий компілятор

КС – комп’ютерна система

КТД – комплект конструкторсько-технологічної документації

МД – монолітні деталі

ММ – математичні моделі

МНІ – машинні носії інформації

МОО – мова опису об’єктів

МОЗ – мова опису задач

НСІ – нормативно – довідкова інформація

ОА – обчислювальний алгоритм

ОМ – об’єкт моделювання

ОП – об’єкт проектування

ОС – операційна система

ОУ – об’єкт управління

ПВВИ – пристрій введення - виведення інформації

ПЗ – програмне забезпечення

ПК – персональний комп’ютер

ПМГ - пристрій машинної графіки

ПОС - пристрій облаштування оперативного зв’язку

ППД - пристрій підготовки даних

ППП – пакет прикладних програм

ПРОПМ – проблемно-орієнтований програмний модуль

РЕА – радіоелектронна апаратура

САВПР – система автоматичного проектування

САПР – система автоматизованого проектування

СО – складальна одиниця

СППР – системи підтримка ухвалення рішень

ССАПР – систем наскрізного автоматизованого проектування

СУБД – система управління базами даних

СУЗ – система управління знаннями

СХЕ – схема електрична принципова

ТЗ – технічне завдання

ТР – технічне рішення

ТД – технічна документація

УП – управляючі програми

УПД – уніфіковане подання даних

ЦБД – централізована БД

ЦБнД – централізований банк даних

Література

1. Аналоговые и цифровые интегральные микросхемы. Справочное пособие /Под ред. С.В.Якубовского. – 2-е изд. перераб. и доп. – М.: Радио и связь, 1985. – 432 с.

2. Белов В.С. Учебное пособие «Информационно-аналитические системы». /Московский международный институт эконометрики, информатики, финансов и права/. М., 2003 г. – 70 с.

3. Галузинський Г. П., Гордієнко І. В. Сучасні технологічні засоби обробки інформації: Навч. посібник. - К.: КНЕУ, 1998.- 134 с.

4. Кальянов Г. Н. -CASE-технологии. Консалтинг в автоматизации бизнес-процессов. – 3-е изд. – М.: Горячая линия-Телеком,2002. -320 с: ил.

5. Кальянов Г. Н. CASE-структурный системный анализ (автоматизация и применение).- М.: ЛОРИ, 1996.- 242 с.

6. Каземіров Г.Г., Соколов А.Г., Основи побудови САПР і АСТПП.-М..: Вища школа, 1989.- 294с7. Кондаков А.И. САПР технологических процессов: учебник для студ. высш. учеб. заведений. – М.: «Академия», 2007. – 272 с.

8. Корнейчук В.И., Тарасенко В.П., Мишинский Ю.Н. Вычислительные устройства на микросхемах: Справочник – К.: Техника, 1986. – 264 с.

9. Кузелин М.О., Кнышев Д.А., Зотов В.Ю. Современные семейства ПЛИС фирмы Xilinx. Справочное пособие. - М.: Горячая линия-Телеком, 2004. – 164 с.

10. Курейчик В.М. Математическое обеспечение конструкторского и технологического проектирования с применением САПР, учебник для вузов, - М.; Радио и связь, 1990, - 352 с.

11. Ли К. Основы САПР (CAD/CAM/CAE). – СПб.: Питер, 2004. – 560 с.: ил.

12. Лопаткин А.В. Проектирование печатных плат в системе P-CAD: Учебное пособие для практических занятий. – Нижний Новгород: НГТУ, 2002. – 178с.

13. Малюх В.Н. Введение в современные САПР: Курс лекций. – М.: ДМК Пресс, 2010. – 192 с.: ил.

14. Мартин Дж. Системный анализ передачи данных. – М.: Мир, 1975 – 256 с.

15. Морозов К. К., Одиноков В. Г. Курейчик В. М. Автоматизированное проектирование конструкций радиоэлектронной аппаратуры, - М.; Радио и связь, 1983, - 280 с.

16. Николайчук Я.М., Возна Н.Я., Пітух І.Р. Проектування спеціалізованих комп'ютерних систем /Навчальний посібник. – Тернопіль: ТзОВ «Терно-граф», 2010. – 392 с.

17. И.П.Норенков Автоматизированное проектирование: М.: Изд-во МГТУ им.Н.Э.Баумана, Москва – 2000 - 188с.

18. Норенков И. П. Введение в автоматизированное проектирование технических устройств и систем: Учебн. пособие для втузов.- М.: Высш. шк., 1986.- 304 с.

19. Палагин А.В., Яковлев Ю.С., Системная интеграция компьютерной техники: Монография. – Винница: УНИВЕРСУМ, 2005 – 680 с.

20. Петров С.В. Шины PCI, PCI Express. Архитектура, дизайн, принципы функционирования. - БХВ-Петербург, 2006– 204 с.

21. Г.Г. Казеннов, А.Г. Соколов «Основи побудови САПР і АСТПП», М., Вища школа, 1989г. – 295 с.

22. Петухов О.А. Моделирование: системное, имитационное, аналитическое: учеб. Пособие / О.А.Петухов, А.В.Морозов, Е.О.Петухова. доп-е изд.: Изд-во СЗТУ, 2008. - 288 с.

23. Саврушев Э.Ц. P-CAD для Windows. Система проектирования печатных плат. Практ. пособие. – М.: ЭКОМ, 2002. – 320 с.

24. САПР Кн.3 Информационное и прикладное программное обеспечение. Федорчук В.Г., Черненький В.М. под ред. Норенкова И.П., М., Изд-во МГТУ им.Н.Э.Баумана, 1986– 274 с.

25. Системы автоматизации проектирования и производства //Радио-электроника (состояние и тенденции развития). Сер. 1. Вычисл. техника.- М.: НИИ экономики и информ. по радиоэлектронике (НИИЭИР), 1985.- 47 с.

26. Системы автоматизированного проектирования. Кн.1.(Под ред. Норенкова И.П.) М., 1986– 164 с.

27. Справочник по САПР. Под ред. Акад. АН УССР В.И.Скурихина. К., «Техника», 1988, - 376 с.

28. Столлингс В. Структурная организация и архитектура компьютерных систем. Пер. с англ. – М.: Издательский дом «Вильямс», 2002 – 196 с.

29. Уваров А.С. PCAD 2002 и SPECCTRA. Разработка печатных плат. – 2-е изд. испр и доп. – М: СОЛОН-Пресс, 2005. – 544 с.

30. Уваров А.С. P-CAD. Проектирование и конструирование электронных устройств. – М.: Горячая линия-Телеком, 2004. – 756 с.

31. Федотова Д.Э., Семенов Ю.Д., Чижик К.Н. CASE-технологии: Практикум. - М.: Горячая линия-Телеком, 2005.- 160 с: ил.

Навчальний посібник

Савеленко Олена Костянтинівна
Якименко Наталія.Миколаївна
Колодочкіна Аліса Володимирівна
Сорокін Володимир Володимирович

ТЕХНОЛОГІЇ ПРОЕКТУВАННЯ КОМП'ЮТЕРНИХ СИСТЕМ

Українською мовою

Редактор: Савеленко О.К.

Формат 60x84/16. Ум. друк. арк. 19,25.
Обл. вид. арк. 17,3. Тираж 100 пр. Зам. № 96.

Видавець і виготовлювач Лисенко В.Ф.
вул. Пацаєва, 14, корп. 1, кв. 101. м. Кропивницький,
Україна, 25030

Свідоцтво суб'єкта видавничої справи ДК №3904 від 22.10.2010