МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Кафедра залізничного, автомобільного транспорту
та підйомно-транспортних машин

Кузьменко С.В.

КОНСПЕКТ ЛЕКЦІЙ
З ДИСЦИПЛІНИ «ОСНОВИ ОБ'ЄКТНО-ОРІЄНТОВАНОГО ПІДХОДУ ПРИ РОЗРОБЦІ ТЕХНІЧНИХ СИСТЕМ»

ЗАТВЕРДЖЕНО
на засіданні кафедри ЗАТПТМ
Протокол №10 від 20.05.2024 р.

Київ, 2024

УДК 629.4

Конспект лекцій з дисципліни «Основи об’єктно-орієнтованого підходу при розробці технічних систем» (для студентів спеціальності Залізничний транспорт) \ Укл. Кузьменко С.В. – Київ: Вид-во Східноукраїнського національного університету ім. В. Даля, 2024.- 183 с.

Конспект лекцій призначений на допомогу студентам, які навчаються за освітньою програмою - Залізничний транспорт, освітнього рівня - магістр. Для студентів денної та заочної форми навчання.

Укладач С.В. Кузьменко, проф.

Відповідальний. за випуск А.О.Климаш, доц.

Рецензент О.В. Фомін, проф.

ЗМІСТ

1. ОСНОВИ ТЕОРІЇ СИСТЕМ

1.1. Основні поняття і визначення в теорії систем

Поняття "система" походить від грецького слова SISTEMA, що означає "ціле, яке складене з частин, або з'єднання (сполучення)". Сучасне поняття системи сформульовано в рамках загальної теорії систем. Під ним розуміють множину елементів, які знаходяться у відношеннях і зв'язках між собою й утворюють деяку цілісність, єдність для досягнення поставленої мети. В цьому випадку мета являє собою сукупність результатів, що визначаються призначенням системи. Наявність мети змушує зв'язувати елементи в систему, тобто виникає потреба цілісності - найбільш важливої властивості системи. Елемент належить системі тому, що він зв'язаний з іншими її елементами, так що множину елементів, які складають систему, неможливо розділити на деяку підмножину. Усунення із системи елемента або сукупності елементів призводить до зміни її властивості в напрямі, відмінному від мети. Термін "система" з'явився в науковій літературі давно. Найбільш широко цей термін використовувався в механіці, де позначав матеріальну систему, тобто сукупність матеріальних точок, які підпорядковані деяким зв'язкам. Основний інтерес для цих систем становлять задачі динаміки, які виявляють причинно-наслідковий механізм їх руху. Закони динаміки (або за сучасною термінологією — закони функціонування механічних систем) були отримані довгим індуктивним шляхом. Висунуті гіпотези перевірялись на багаточисленних дослідах. Деякі з цих дослідів стали класичними. Перевірялися також багаточисленні наслідки висунутих гіпотез. Усе це було реалізовано завдяки існуючій в механіці, а також у більшості інших розділів фізики можливості ставити "чисті досліди", тобто усувати шкідливі фактори. Крім того, умови могли бути відтворені з досить високою точністю в інший час і в іншому місці.

Останнім часом у літературі досить широко використовуються три

"системних" поняття: "системний аналіз", "системний підхід" і "теорія систем". Між ними досить часто ставлять знак тотожності, що призводить до деякої плутанини. Оскільки в подальшому ми будемо говорити про теорію систем, то нам необхідно чітко визначити всі ці терміни. Слово "система" та пов'язані з ним терміни отримали останнім часом значне поширення. Це пояснюється тим, що все частішою стає потреба вивчення складних комплексів (систем). Така необхідність визначається різким ускладненням створюваних технічних конструкцій, пристроїв, технологій і всієї сукупності господарських зв'язків, з якими доводиться мати справу економістам та господарським керівникам. Потреба вивчення біологічних об'єктів і проблем екології, які з кожним роком стають все більш та більш актуальними, також приводить до дослідження складних систем.

З необхідності вивчення складних систем виникла дисципліна "Системний аналіз", яка є продовженням дослідження операцій [1] в умовах невизначеності. Бувають випадки, коли взагалі не вдається поставити цілі або ті цілі, які ми бажаємо поставити, нереальні. Приклади подібних ситуацій дає нам економіка, коли плануються одні показники, а в дійсності маємо зовсім інші. В цьому випадку необхідно мати систему моделей, створити такий математичний апарат їх аналізу, який дав би змогу реально прогнозувати ті чи інші наслідки наших рішень, оцінити наші можливості при різних альтернативах і тільки на основі такого системного аналізу сформулювати цілі. Складність систем, що вивчаються або проектуються, приводить до необхідності створення спеціальної, якісно нової техніки дослідження, яка називається системами імітації — спеціально організованими системами математичних моделей. Ці моделі за допомогою ЕОМ відображають функціонування комплексу, що проектується або вивчається.

Дослідження динаміки того чи іншого процесу, яке дає можливість побачити перспективи і намітити цілі, — це лише один з аспектів системного аналізу. В рамках системного аналізу вивчаються проблеми проектування ієрархічної організації. Будь-які більш-менш складні системи завжди

організовані за ієрархічним принципом, бо централізовані обробка інформації та прийняття рішень досить часто бувають неможливими через великий обсяг інформації, яку треба збирати і переробляти. В проектуванні технічних систем задача системного аналізу (задача проектувальника) полягає, перш за все, в розробленні самої функціональної схеми (яка може бути реалізована не єдиним способом) і у визначенні окремих цілей системи.

По відношенню до технічних систем значно складнішими системами є народногосподарські комплекси, функціональні елементи котрих залежать від того, як керують ними люди. На відміну від машини людина завжди має власні цілі й інтереси, і проектувальнику системи вже недостатньо тільки формулювати цілі для нижніх ланок. Необхідно ще бути впевненим, що ці цілі будуть досягнуті, тобто що нижні ланки виконають вимоги верхніх ланок. А для цього, в свою чергу, повинна бути запроектована спеціальна система.

Теорія ієрархічних систем, яка займається деякими аспектами цієї проблеми, є однією з найважливіших частин системного аналізу.

Таким чином, системний аналіз — це технічна дисципліна, що розвиває методи проектування складних технічних і народногосподарських систем, організаційних структур і т.д. Поряд із терміном "системний аналіз" значне поширення отримав інший термін - "теорія систем". Виникнення "теорії систем" пов'язують з іменем відомого біолога Людвіга фон Бертеланфі [2], який у п'ятдесятих роках в Канаді організував центр системних досліджень й опублікував велику кількість робіт, у тому числі і книг, у яких намагався знайти те загальне, яке притаманне будь-яким досить складним організаціям матерії як біологічної, так і соціальної природи. Однак значно раніше російський учений А. А. Богданов створив теорію організації [3]. У своїй роботі А. А. Богданов увів поняття організації як одне з початкових понять. Матерія існує в часі та в просторі. Вона завжди має ту або іншу організацію. В той же час і організацію неможливо уявити без матеріального носія. Основу для побудови теорії А. А. Богданов бачив у тому, що, незважаючи на

фантастичну різноманітність матеріалу, який існує в природі, кількість архітектурних або організаційних форм відносно невелика. Виходячи з цього, теорію систем можна трактувати як методологію науки, що є загальною теорією.

Крім того, існує поняття "системний підхід", яке відображає деякі тенденції в створенні систем. У розвитку науки завжди пролягають дві лінії - аналіз і синтез. Ми завжди бачимо прагнення до аналізу — вивчення конкретних фактів, проникнення в глибину явища, яке вивчається, розкриття структури того чи іншого явища і т.д. Поряд із цим завжди існує прагнення створити синтезуючі теорії, які дозволяють об'єднати різні факти, побачити перспективи того чи іншого процесу, його зв'язки з іншими явищами тощо. У різні періоди часу значення підходів було різним. Останнім часом, коли на людство "навалилась" лавина нових фактів, увага до синтезуючих побудов стала особливо важливою. Потреба не просто вивчати явище або факт, а встановлювати його зв'язок з іншими фактами і привела до появи спеціального терміна "системний підхід", Дослідник завжди прагнув по можливості системно підходити до вивчення того або іншого явища. Однак він не завжди міг мати в своєму розпорядженні необхідний інструмент. Тепер в епоху ЕОМ ці можливості різко зросли. Звідси, як наслідок, і прагнення до вивчення явища в усій його повноті, у зв'язку з іншими явищами. Системний підхід безперервно стимулюється потребами практики, яка висуває все більш складні проекти, що вимагають аналізу міжгалузевих та міждисциплінарних проблем.

З аналізу розглянутих системних понять можна зробити висновок, що найбільш загальним теоретичним терміном є теорія систем, яка є основою для більш практичних понять: системний аналіз і системний підхід. Тому теорію систем можна вважати базовою теоретичною дисципліною, що свої основні положення реалізує в більш практичних дисциплінах: "Системний аналіз" і "Системний підхід". У зв'язку з цим, далі ми будемо мати справу тільки з теорією систем.

1.2. Класифікація систем

Класифікацію систем можна проводити за різними ознаками. На сьогодні дати чітку завершену класифікацію систем неможливо, бо науки, що займаються дослідженням систем, знаходяться в стадії розвитку. До цих наук належить і теорія систем, яка спрямована на розроблення загальних теоретичних положень моделювання систем, їх дослідження з метою визначення властивостей та можливих варіантів поведінки, створення систем для виконання необхідних функцій із заданими властивостями і керування з метою забезпечення необхідної поведінки систем.

Виходячи із загального розгляду систем, останні можна розділити на два великих класи: матеріальні й абстрактні.

Матеріальні системи поділяються на системи неорганічної природи (фізичні, хімічні, геологічні, технічні і т.д.) та живі системи (найпростіші біологічні системи, організми, популяції, види, екосистеми). Особливий клас матеріальних живих систем — це соціальні системи (від найпростіших соціальних об'єднань до соціально-економічної структури суспільства).

Абстрактні системи — це поняття, гіпотези, теорії, наукові знання про системи, лінгвістичні, логічні та інші системи.

Крім розглянутої класифікації, системи можуть бути розділені за функціональними і просторовими ознаками, типам складності системи, видом моделювання тощо.

Систему виділяють із зовнішнього світу або за просторовими, або за функціональними ознаками. Система, як правило, має просторову чи функціональну замкненість. Це означає, що можна провести межу або в просторі компонентів цієї системи, або в просторі її функцій, з одного боку якої знаходиться система, а з іншого — зовнішнє середовище. При цьому властивості системи відмінні від властивостей зовнішнього середовища.

Наведемо декілька прикладів систем: 1) Сонячна система; 2) живий

організм; 3) обчислювальний центр; 4) промислове підприємство; 5) гідравлічна схема; 6) карний кодекс держави; 7) система лінійних рівнянь; 8) галузь промисловості; 9) система соціального забезпечення; 10) операційна система ЕОМ; 11) автоматизована система керування технологічним процесом; 12) система планування народного господарства; 13) серцево-судинна система. Системи 1-7 складаються з матеріальних й абстрактних об'єктів і сформовані за просторовими ознаками, системи 8 - 13 — за функціональними ознаками. Деякі з перерахованих систем допускають двоякий опис. Так, операційна система може задаватись як своїми функціями (керування процесом проходження задач та розподілення ресурсів даної ЕОМ), так і набором програм, які реалізують ці функції.

У тих випадках, коли система задається просторовими ознаками, дослідник однозначно проводить структуризацію системи. Під структуризацією розуміють виділення в системі двох типів об'єктів — множини елементів і множини зв'язків — та встановлення співвідношень цих множин між собою. Так, основними елементами Сонячної системи є Сонце й планети, а зв'язками — гравітаційна взаємодія між ними. В промисловому підприємстві елементами можуть бути окремі цехи, а зв'язками — матеріальні, енергетичні й інформаційні потоки між ними. В системі лінійних рівнянь елементи — це окремі рівняння, а зв'язки — участь одних і тих незмінних у різних рівняннях.

У рамках однієї і тієї ж системи структуризація може бути проведена по-різному. Так, структурною одиницею (елементом) підприємства може бути як цех, так і дільниця або робоче місце. Відповідно до прийнятої структуризації змінюються види зв'язків. Крім того, те, що в одних випадках виступає як вид зв'язку, в іншому може вважатися видом елемента. Наприклад, муфти привода будь-якого кранового механізму можуть розглядатись не як елементи, а як зв'язки між окремими елементами (двигуном і редуктором, редуктором та барабаном і т.д.). Системи можна розділити на складні і прості. На сьогодні не існує достатньо загального

визначення складної системи. Тому, залежно від типу об'єкта дослідження, використовується те або інше поняття складної системи, яке справедливе для одного об'єкта, але не завжди справедливе для іншого.

До характерних особливостей складних систем можна віднести [6]: 1) велику кількість взаємозв'язаних між собою елементів і підсистем;

2) складність функцій, що виконує система для досягнення мети її функціонування;

3) багатомірність системи, яка зумовлена наявністю великої кількості зв'язків між підсистемами;

4) взаємодія із зовнішнім середовищем і функціонування в умовах дії випадкових факторів;

5) наявність великої кількості критеріїв оцінки якості функціонування системи та її підсистем;

6) багатоманітність структури складної системи, яка обумовлена як різноманітністю структур її підсистем, так і різноманітністю структур об'єднання підсистем в єдину систему;

7) наявність керування, яке має ієрархічну структуру, а також розгалуженої інформаційної мережі й інтенсивних інформаційних потоків;

8) багатоманітність фізичної природи підсистем, які характеризуються їх різною фізичною сутністю;

9) велика розмірність і складність моделі системи;

10) існування ознак, які притаманні системі в цілому, але не властиві кожному окремому елементу (наприклад, резервована система надійна, а її елементи можуть бути ненадійними; замкнена система, що складається зі стійких елементів, може бути нестійкою);

11) відсутність можливості отримання достовірної інформації про властивості системи в цілому в результаті вивчення її окремих елементів.

Таким чином, складна система являє собою множину взаємозв'язаних і взаємодіючих між собою елементів та підсистем різної фізичної природи, які складають нероздільне ціле, що забезпечує виконання системою деякої

складної функції (наприклад, забезпечення ритмічного виробництва автомобілів великим заводом).

Система вважається простою, якщо складається з малої кількості елементів однієї фізичної природи, які утворюють нероздільне ціле, що забезпечує виконання системою деякої простої функції (наприклад, перетворення обертального руху в поступальний або навпаки).

1.3. Будова, функція і структура системи

Будова системи. Розчленування системи на елементи є одним *з* перших кроків при побудові її формального опису. При цьому елемент виступає як об'єкт, який при даному розгляді системи не підлягає подальшому розчленуванню на окремі частини. Таким чином, елемент — це мінімальний неподільний об'єкт.

Неподільність елемента — це поняття, але не фізична властивість. керуючи поняттям "елемент", дослідник залишає за собою право перейти на інший рівень розгляду питань і досліджувати самі елементи та їх склад, а це свідчить про фізичну роздрібленість елементів. Таким чином, об'єкти називаються елементами за домовленістю, яка приймається з метою дати відповідь на конкретні питання, що стоять перед дослідниками. завдань дослідження може вимагати розчленування елементів на неповні частини або об'єднання декількох елементів в один. Так, при дослідженні надійності роботи вантажопідйомного крана елементами системи вважаються окремі агрегати механізмів: двигуни, гальма, редуктори, муфти, барабани і т.д., а для дослідження кінематики руху вантажу окремими елементами можуть бути механізми підйому та зміни вильоту вантажу, повороту та переміщення крана, до складу яких входять окремі агрегати. Тобто в другому випадку здійснено об'єднання декількох агрегатів в один механізм на функціональній основі, і система крана представлена у вигляді чотирьох елементів (механізмів). У ряді випадків, наприклад при дослідженні

міцнісних характеристик крана, агрегати останнього розчленовуються на окремі деталі. При цьому кількість елементів системи значно збільшується, зв'язки між елементами ускладнюються. Таку систему можна вже віднести до складної системи. Для кожного елемента тієї чи іншої системи характерні такі властивості, які визначають його взаємодію з іншими елементами або впливають на властивості системи в цілому.

У ряді випадків складні системи розділяються за просторовими або функціональними ознаками ієрархічно на підсистеми. Такі підсистеми являють собою сукупність елементів. Формально будь-яку сукупність елементів системи з їх зв'язками можна розглядати як підсистему. Однак використання цього поняття найбільш ефективне, якщо підсистема є достатньо самостійною частиною складної системи, але мета її функціонування підпорядкована загальній меті функціонування системи. Розчленування складних систем на підсистеми називається декомпозицією систем. Цей процес поки що не формалізовано, і він має еврістичний характер. Правильне виділення підсистем дозволяє спростити й полегшити процеси моделювання, аналізу, синтезу та керування систем. Розглянемо приклад. Нехай система складається з $n = 20$ елементів, між якими існує л(и-1)=380 зв'язків. Розчленуємо (якщо це можливо) систему на 4 підсистеми по 5 елементів у кожній (и/=5). Тоді число зв'язків між елементами однієї підсистеми - й;(и;-1)=5'4=20, а в чотирьох підсистемах буде 20 ' 4 = 80 міжелементних (внутрішніх)

зв'язків. Підсистеми мають між собою 4 • 3 = 12 зв'язків. Таким чином, у розчленованій системі можна розглядати всього 80 + 12 = 92 зв'язки, що значно менше ніж у нерозчленованій системі. Тому створення, дослідження, передбачення поведінки і т.д. таких систем значно спрощується.

Функція системи та її структура. Реальні системи описують шляхом визначення їх функцій і структур.

Функція системи — це правило отримання результатів, передбачених метою (призначенням) системи. Визначаючи функцію системи, її поведінку

описують шляхом використання деякої системи понять: відношення між змінними параметрами, векторами, множинами. Функція встановлює, що робить система для досягнення поставленої мети безвідносно до фізичних засобів (елементів, зв'язків), що складають саму систему і не визначають, яким чином побудована система. Системи вивчають на різних рівнях абстракції, з використанням різних підходів, кожний із яких дає відповідь на ті чи інші питання. В зв'язку з цим, функції системи можуть описуватися з різним ступенем деталізації. Для опису функцій системи використовуються теорії множин, алгоритмів, випадкових процесів, інформації. Якщо система функціонує, то це значить, що вона отримує результати, передбачені призначенням системи.

Реальні складні системи функціонують в умовах дії великої кількості випадкових факторів, котрі можуть бути як зовнішніми, так і внутрішніми (наприклад, шуми, вібрація, зміна температури, радіація і т. д.). Ці фактори ускладнюють функціонування складних систем, а тому повинні враховуватись при їх створенні і досліджуватися в процесі експлуатації систем.

Структура системи — це фіксована сукупність елементів і зв'язків між ними. В загальній теорії систем під структурою прийнято вважати тільки множину зв'язків між елементами. Тобто під структурою розуміють деяку картину, яка відображає тільки конфігурацію системи безвідносно до складових її елементів. Таке тлумачення її поняття зручне за структурним підходом при вивченні властивостей різних систем — систем із паралельними, послідовними, змінними, ієрархічними структурами і зворотними зв'язками (рис. 1.1).

На практиці поняття "структура" включає в себе не тільки множину зв'язків, але й множину елементів, між якими існують зв'язки. Найбільш часто структура системи зображається у формі графа: елементи системи представляються вершинами графа, а зв'язки — дугами (ребрами) графа (рис. 1.2).

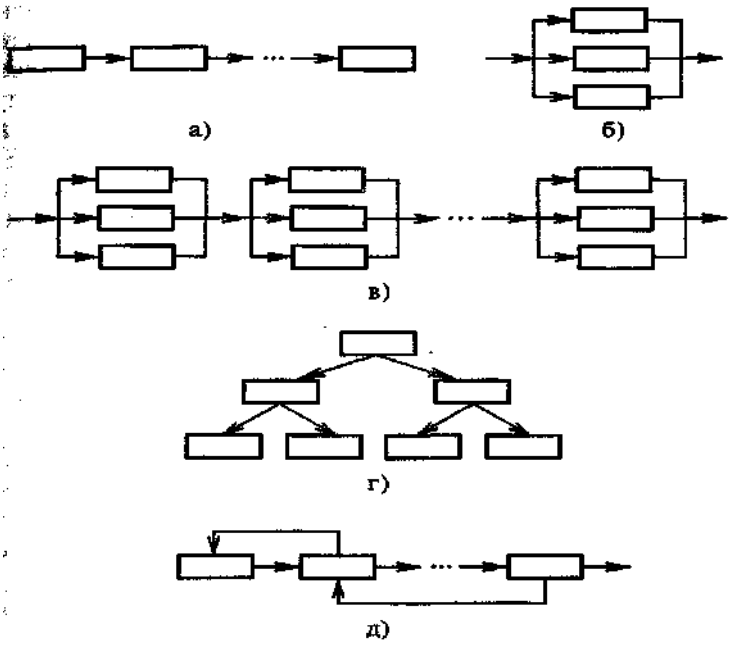

Рис. 1.1. Системи з послідовними (а), паралельними (б), змішаними (в), ієрархічними (г) і зворотними (д) зв'язками елементів

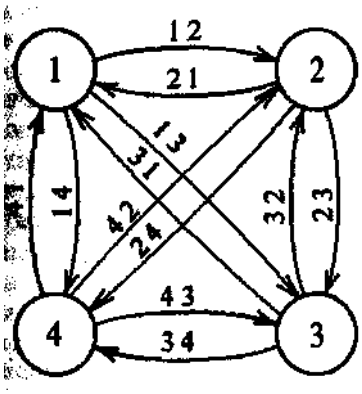

Рис. 1.2. Зображення структури системи у формі графа: 1,2,3,4 - елементи системи; 12, 21, 13, 31, 14, 41, 23, 32, 24,42, 34, 43 - зв'язки між елементами системи

Організація складних систем являє собою процес упорядкованого розміщення множини елементів з урахуванням їх логічних зв'язків із метою здійснення необхідних функцій у складних системах. Звичайно, до однієї і тієї ж мети можна прийти різними способами, виходячи з різних принципів організації систем. Кожний принцип організації задає деякий спосіб побудови множини систем, аналогічних за призначенням, але різних за

функціями й структурами. Конкретна система становить собою лише приклад реалізації деякого способу організації. Наприклад, більшість сучасних ЕОМ будується на основі одного принципу організації — принципу програмного керування реалізації алгоритма на основі команд, які мають операційно-адресну структуру. Таким чином, організація — це поняття більш високого рангу, ніж функція і структура. Це — модель, на основі якої можуть будуватися різні конкретні системи.

Будь-який спосіб побудови функцій, достатніх для досягнення деякої мети (сукупності результатів), називають способом *функціональної організації.* Спосіб побудови структури складної системи з набору елементів, яка забезпечує реалізацію функцій необхідного класу, називають способом *структурної організації.* Визначаючи спосіб функціональної організації, виявляють клас функцій, притаманних системам певного призначення (безвідносно до засобів, необхідних для реалізації цих функцій), а визначаючи спосіб структурної організації, виявляють правило побудови структур, що реалізують виявлений клас функцій, які відповідають певному призначенню.

Керування являє собою процес збору, обробки та передачі інформації.

У системі виділяють контури керування, вздовж яких циркулюють потоки інформації. Від елементів керування до пристроїв керування поступає для обробки первинна інформація, а останні видають інформацію для керування.

Існує достатньо великий клас самоорганізуючих (самоналаджуючих) систем, які здатні в результаті дії зовнішнього середовища перейти шляхом послідовної зміни своїх властивостей до деяких стійких станів. До таких систем належать, наприклад, механічні системи зі сталою швидкістю руху, які обладнані регулятором Уатта.

1.4. Предмет теорії систем

Теорія систем являє собою аксіоматичну математичну теорію, в рамках якої розроблено концептуальний апарат і ефективні методи дослідження систем довільної природи. В теорії систем дається математичне визначення її предмета. Це визначення базується на формалізації зв'язків між елементами системи. Якщо формалізовано

уявлення про зв'язок між двома елементами, формальний опис системи взаємозв'язаних (взаємодіючих) елементів $A_1, \ldots, A_K$ перетворюється на композицію таких формальних зв'язків між відповідними парами елементів (A_1, A_j). Зв'язок завжди означає взаємодію елементів. У результаті приходимо до висновку, що система являє собою взаємодіючі в часі процеси.

Трансформація системи привела до математичного визначення її предмета, яке ввів Р. Калман [4]. Він виділив основні поняття і сформулював основні проблеми, з яких виросла, інтенсивно розвивається й знаходить широке практичне застосування сучасна теорія систем.

Для формалізації поняття зв'язку використовуються деякі первинні поняття. Вони відображають деякі сторони організації реальних процесів. Будь-який процес являє собою послідовність у часі реальних явищ. При цьому він не є абсолютно довільною послідовністю: явища якимсь чином об'єктивно організовані. Ця організація, порядок і є змістом поняття "система". В найбільш загальному вигляді цей порядок установлюється двома принципами діалектичного матеріалізму: детермінізмом і причинністю.

"Усі форми реальних взаємозв'язків явищ у кінцевому рахунку складаються на основі загальнодіючої причинності, поза якою не існує жодне явище дійсності... Причинність загальна, бо немає явищ, які не мали б своїх причин, так і немає явищ, які не породжували б або інших наслідків" [5]. Філософське розуміння детермінізму пов'язане з поняттям причинності. "Центральним ядром детермінізму служить положення про існування

причинності, тобто такого зв'язку явищ, в якому одне явище (причина) при досить певних умовах з необхідністю породжує інше явище (наслідок)".

При математичному формулюванні цих положень поняття причинності відображається в понятті стану і властивостей закону зміни стану, а поняття "певні умови" — в понятті входу. Причинно-наслідковий зв'язок, який задовольняє принципи детермінізму і причинності, означає, по-перше, що жодне реальне явище не виникає спонтанно, довільно, завжди є передуюче йому в часі, інше реальне явище, яке його викликає. По-друге, жодне явище, яке реалізувалось у даний момент часу, не залежить від того, які реальні явища пройдуть в моменти часу, що настануть за вказаним (у теорії систем цю властивість називають причинністю). Та обставина, що в даний момент реалізувалося певне явище, а не якесь інше, вказує на наявність певних основ для реалізації саме цього явища. В цьому виражається принцип детермінізму реальних процесів.

Неможливість довільного виникнення явища приводить при формалізації закономірності поведінки процесу до необхідності введення іншого процесу, який знаходиться з даним у причинно-наслідковому зв'язку. В теорії систем причинний процес називають входом, а процес-наслідок — виходом.

Іншим фундаментальним поняттям теорії систем є поняття стану. В літературі, присвяченій теорії систем, концепції стану надається особлива увага. З одного боку, будь-яка природничо-наукова теорія використовує це поняття, а з іншого — ніхто не бачив і не виміряв стан реальних об'єктів (процесів). Експериментальне встановлено, що фізичні властивості об'єктів змінюються залежно від стану, й ці властивості завжди можна ідентифікувати. Однак сам стан завжди скритий. Наявність стану можна обгрунтувати різними способами. Реальна система завжди включає два процеси, один з яких залежить від іншого. Разом з тим при формальному аналізі характеру залежності виходу від входу виявляється, що безпосереднього зв'язку між ними немає. Дійсно, реальна подія в момент часу t не може залежати від того, що в цей момент реально не існує. Події, які

пройшли в процесі-вході в моменти t, передуючи моменту t, у момент t не є реальністю. Тому подія, що являє собою конкретне значення виходу в момент t, не залежить від значень входу в моменти $T < t$. Разом з тим вихід у момент t також не залежить від входу, який реалізується в той самий момент t, оскільки вплив одного явища на інше не може бути миттєвим, розповсюдження сигналу завжди проходить з кінцевою швидкістю. Причина і наслідок неможуть виникати одночасно.

З одного боку, вихід залежить від входу, а з іншого — не залежить. Розв'язання цього протиріччя полягає в тому, що залежність виходу від входу є опосередкованою. Це показує, що існують об'єкти, які зв'язують усю попередню історію входів-причин до моменту t і вихід у цей момент. Такі об'єкти називають станами. Таким чином, конкретною причиною явища в процесі-виході, основою реалізації саме цього явища слід вважати деякий стан (детермінізм).

У кожний момент t система характеризується деяким станом-елементом її множини станів, який однозначно визначає значення виходу в цей момент t, і це — одна з аксіом теорії систем. Вплив входу на вихід зводиться до залежності стану в кожний момент t від процесу виходу, який реалізується до цього моменту t, тобто в стані накопичуються всі причини, що реалізувалися в минулому і які визначають теперішній стан. Якщо в конструкції поняття системи використання процесів входу і виходу було зумовлено фізичними уявленнями про функціонування системи, то поняття стану має відношення до закону формування виходу. Об'єкт, що взаємодіє з системою, може здійснити цю взаємодію тільки через вхід і вихід системи, а встановити безпосередній зв'язок із процесом у просторі станів неможливо. Знання, в якому стані знаходиться система в деякий момент часу, може бути отримане лише в результаті розв'язування деякої теоретико-системної задачі.

Крім входу, стану і виходу є ще два об'єкти, які необхідні при побудові поняття системи. Поняття системи також містить обмеження на можливі процеси. Це обмеження виражається так званими відображенням виходу й

перехідним відображенням. Оскільки вихід однозначно визначається станом, то існує зв'язок між ними, який виражається відображенням із множини станів у множину значень, що приймаються виходом, яке називається відображенням виходу.

Аналогічно існує зв'язок між входом та станом. Якщо в момент t_0 система характеризувалася станом $x°$, а в момент $t1 > to$ — станом X^n, причому в моменти часу *1,* де $t_0 < T < t1$, вхід приймав деякі значення *и* (т), то зміна стану саме в стан *x′,* а не в будь-який інший, викликається дією визначеного закону поведінки системи. Іншими словами, існує ще одна характеристика — закон, якому підпорядковується поведінка системи в просторі станів. У процесі нормалізації цей закон можна описати у вигляді відображення, яке кожному стану і кожному входу ставить у відповідність якийсь стан, причому це відображення залежить від двох моментів часу й від параметрів системи. Воно називається перехідним відображенням.

Таким чином, конструкція поняття системи включає первинні поняття входу *й (t),* стану *x (t)* і виходу *y (t),* а також відношення між цими поняттями, вираженими відображеннями виходу та переходами (рис. 1.3).

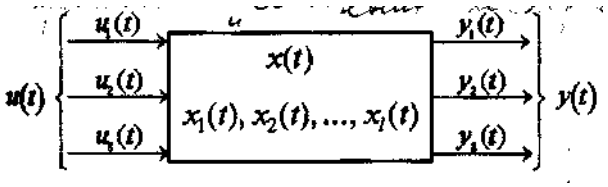

Рис 1.3. Система як перетворювач входу у вихід через свій стан

Знання множини станів, перехідного відображення і відображення виходу дає змогу відповісти на такі питання: 1) яку поведінку може мати система; 2) як необхідно підійти до розв'язування задачі про передбачення поведінки системи; 3) як розв'язати задачу забезпечення необхідної поведінки.

Розв'язування цих задач базується на методології і основних ідеях системного дослідження, які включають у себе питання побудови моделі

систем, які відображають взаємозв'язки реальних ситуацій, їх аналізу, синтезу та керування.

1.5. Основи формалізму теорії систем

Раніше вже відмічалось, що при формалізації системи використовуються три процеси: вхід, вихід і процес у просторі станів. Для математичного завдання процесу необхідно виділити множину його значень та впорядковану множину, що фіксує, в якій послідовності ці значення реалізуються. Досить часто впорядковану множину трактують як час, і тоді мають справу з процесами, що проходять у часі. Впорядкована множина для цих трьох процесів уважається однією й тією ж і називається множиною моментів часу. Вона позначається через Т. Через *U, Y* і *X* позначають відповідно множини значень входу, виходу і множину станів. Значеннями вхідних *u(t)* і вихідних *y(t)* процесів, а також процесів стану *x(t)* системи в деякий момент часу є відповідно елементи множин *U, Y* та *X*: *u(t)єU, y (t)є Y* і *x(t)єX*.

Кожна конкретна система характеризується своєю множиною входів, які є допустимими для цієї системи. Множина всіх реакцій системи, тобто виходів також є характеристикою системи. Конкретний вихід y(t) є *Y* в кожний момент *t* повністю визначається станом і тільки станом системи в цей момент x(t)єX. Тоді існує відображення елементів множини *X* в елементи множини *Y (X-»Y)* таке, що виконується співвідношення .

$$y(t) = \eta(t,\ x(t)),\quad t \in T. \qquad (1.1)$$

Тут залежність відображення n від *t* означає, що характер залежності виходу від стану зі зміною часу може змінитися.

Відсутність залежності виходу *y* в момент *t* від u *(t)* можна також інтерпретувати як неможливість за нескінченно малий час, змінюючи вхідну дію, викликати зміну виходу системи. Разом із тим у теорії дискретних за часом систем відображення n інколи визначають і на множині *U*, тобто співвідношення (1.1) має вигляд

$$y(t_k) = \eta(t_k, x(t_k), u(t_k)), \quad k = 1, 2, \ldots, n.$$

Це пояснюється тим, що час реакції виходу на вхід у деяких ситуаціях безмежно малий по відношенню до інтервалу дискретності, і в моделі його можна не враховувати. Відображення n називається відображенням виходу, або функцією спостереження. Відповідно до аксіом до стану, в кожний момент *t* система знаходиться в певному стані, причому стан у моменти *t >m* однозначно визначається станом у момент *t* і відрізком входу *u(m, t).* У цьому полягає принцип детермінізму (певності) в поведінці систем. При формалізації цієї обставини встановлюється існування набору

відображень елементів множини *U* у множину *X (U-»X)* для всіх значень параметрів *m є T, t є T* і *m<t.* Конкретне відображення, яке відповідає фіксованим *m* і *t,* дозволяє для будь-яких *x* та будь-яких *(τ ,t)* визначити стан у момент t, якщо в момент *τ* система знаходилася в стані *x(τ)* і використовувався вхід u *(τ, t),* за залежністю

$$\bar{x}(t) = \mu_a(x(t), u(\tau, t)). \qquad (1.2)$$

Аксіома однозначної визначеності стану в моменти *t* >т за станом у момент т і входу u *(*т, *t)* накладає обмеження на набір відображень

Згідно з цією аксіомою кожний стан системи однозначно визначає майбутній стан (детермінізм процесів). Виходячи із залежність (1.2), можна побудувати відображення *a,* через яке визначається стан системи

$$x(t) = \sigma(t; \tau, x(\tau), u(\tau, t)). \qquad (1.3)$$

Перехідне відображення а повинно відповідати вимозі, що B=ff(t; *t, x, u)* при всіх *t, x, u,* тобто за проміжок часу нульової довжини система не може перейти в інший стан (або в один і той же мо-мент часу система не може знаходитися в двох різних станах). Крім того, відображення сг повинно бути таким, щоб стан у момент *t* не залежав від значень входу, які поступають у моменти часу, більші від момєнту t.

Виходячи з попередніх залежностей, можна дати математичне визначення системи.

Деяка система Е визначена, якщо задані впорядковані множини

моментів часу T, множини значень входів U, виходів Y і станів перехідне відображення (7, яке задовольняє аксіомам узгодженості, детермінізму й причинності, і відображення виходу n такі, що для будь-якого *y (t)* є Y існує *x* (t) є X і u *(t)* є U, для яких при будь-яких де , виконується співвідношення

$$y(t) = \eta\left(t, \sigma\left(t; \tau, x(\tau), u(\tau, t)\right)\right), \qquad (1.4)$$

і, навпаки, будь-який процес *y (t), t*>т, отриманий відповідно (1.4), належить Y.

Іншими словами, систему, яка породжує процеси виходів з Y, можна визначити трьома величинами g, X, n. Ця трійка є виразом закону поведінки системи, введеним Р.Калманом [4]. Більш традиційним є визначення системи шляхом задання співвідношень, які описують відображення g. Ці співвідношення (рівняння) називають моделлю системи. Однак останнє визначення є менш загальним. Опис системи, наприклад, у термінах диференціальних рівнянь вимагає гладкості відображення g, що в загальному випадку не обов'язкове.

2. ТЕХНІЧНІ СИСТЕМИ

2.1. Основні поняття про технічні системи

Понад три мільярди років триває еволюція живих істот. Близько трьох мільйонів років формувався в них розум, і всього сорок-п'ятдесят тисячоліть становить вік того, хто згодом назвав себе людиною розумною.

Людина завжди дивувалася гармонії природи і вчилась у неї, створюючи дедалі складніші об'єкти, які тепер називають технічними системами.

Технічна система походить від грецького слова "TECHNE" - мистецтво, майстерність. Вона являє собою сукупність засобів, які створюються для здійснення процесів виробництва та обслуговування невиробничих потреб суспільства. Основне призначення технічних систем — повна або часткова заміна виробничих організаційних функцій людини з метою полегшення праці і підвищення її продуктивності. Розрізняють технічні системи - виробничі (машини, механізми, інструменти, апаратура керування машинами і технологічними процесами, засоби транспорту, зв'язку, окремі підприємства, виробничі комплекси тощо) й невиробничі (побутові, комунальні, наукових досліджень, освіти, культурні, військові, медичні тощо).

Залежно від принципу дії і будови базових елементів, які здійснюють передачу та перетворення енергії й інформації, технічні системи поділяються на механічні, гідравлічні, електричні, електронні, оптичні. Більшість технічних систем, як правило, являють собою змішані системи, де використовують елементи різної фізичної природи, наприклад: електромеханічні, гідромеханічні, оптико-механічні та інші технічні системи.

Серед розглянутих систем особливе місце займають механічні системи. Справа у тому, що практично всі складні технічні системи мають в своєму складі або елементи механічних систем, або цілі підсистеми механічних систем.

Під механічною системою розуміють довільний пристрій, механізм, машину, конструкцію тощо, у яких передача й перетворення енергії здійснюється за допомогою механічних елементів (валів, зубчастих коліс, муфт і т. ін.). В ряді випадків до механічної системи можуть бути зведені окремі системи живих істот, наприклад, опорно-рухальна система тварин та людини може бути представлена у вигляді шарнірно-важільного механізму.

Ряд інших об'єктів також може бути змодельовано у вигляді механічних систем. Ці системи досить прості, достатньо наочні і найбільш досліджені.

Розглянемо деякі технічні системи різних рівнів галузі будівельного виробництва.

2.2. Виробничо-організаційна технічна система

Розглянемо будову і характеристики виробничо-організаційної технічної системи на прикладі системи будівництва греблі гідроелектростанції (ГЕС). Ця система служить для забезпечення та керування процесом будівництва й складається з таких господарств (підсистем) (рис. 2.1): кар'єри, залізобетонні заводи, транспортні лінії, укладальний комплекс греблі, постачальний комплекс обладнання, комплекс монтажних і підйомно-транспортних робіт. При будівництві ГЕС найбільш трудомісткими та тривалими є бетонні роботи, тому при розгляді цієї системи найбільшу увагу будемо звертати саме на них.

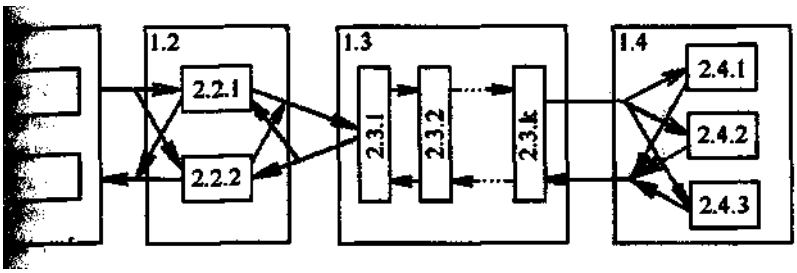

Рис. 2.1. Технічна система будівництва греблі ГЕС:

Система в цілому працює в такому порядку. Добутий у кар'єрах камінь переробляється в щебінь спеціально підібраною системою машин і відправляється на бетонні заводи для приготування бетону. З інших кар'єрів доставляється пісок, який також використовується для приготування бетону. Для різних робіт існують різні марки бетону, а кожний із заводів має можливість готувати будь-яку з марок. Крім того, на залізобетонних заводах виготовляють необхідні залізобетонні й арматурні конструкції. Кожний з бетонозмішувальних вузлів заводу може готувати будь-яку з марок бетону (марки бетону відрізняються процентним вмістом цементу і добавок), однак для переналагоджування виробництва з однієї марки на іншу витрачається певний час. Бетон готується кожним із бетонозмішувальних вузлів порціями, які завантажуються в пересувні бетонозмішувачі і по трасі відвозяться на місце будівництва. Розвантажування, укладка та ущільнення бетону проводяться в різних місцях різними машинами (бетоноукладачами, бетононасосами, вантажопідйомними кранами тощо) і за різними технологіями, які залежать, крім всього іншого, від етапу будівництва. Із збільшенням висоти греблі змінюється склад машин для укладання і ущільнення бетону, умови під'їзду до них. Розвантажені бетонозмішувачі їдуть у зворотному напрямку на бетонні заводи.

Підсистеми 1-го рівня:

1.1 - кар'єри; 1.2 - бетонні заводи; 1.3 - транспортні зв'язки; 1.4 - комплекси для будівлі греблі.

Підсистеми 2-го рівня: 2.2.1,2.2.2 - 1-ий і 2-ий бетонні заводи; 2.3.1 - 2.3.К - ділянки шляху; 2.4.1 - 2.4.3 - розвантажувальні пристрої

В арматурних цехах готуються арматурні конструкції, які частково спеціальними транспортними засобами відправляються на будівництво греблі, а інша частина металоконструкцій використовується для виготовлення залізобетонних конструкцій. Ці конструкції виготовляються на спеціальних постах, де у форму укладається арматура і бетонна суміш. Після ущільнення й твердіння бетону в пропарочних камерах залізобетонні

конструкції відправляються на будівництво греблі спеціальними транспортними засобами. Після розвантажування та монтажу залізобетонних конструкцій підйомними кранами транспортні засоби повертаються на бетонні заводи, і цикл їх роботи повторюється. Щоб мати уявлення про розміри задач, які треба розв'язувати, скажемо, що на будівництві працює декілька заводів, які виготовляють біля 40 марок бетону і декілька десятків типів залізобетонних й арматурних конструкцій; у кар'єрах працюють технологічні лінії із системи машин на виготовлення щебеню (бурові машини, дробарки, грохоти, транспортери, навантажувачі) і для добування піску (екскаватори й автомобілі; автотранспортні підприємства (управління механізації) нараховують декілька сот автомобілів та декілька десятків будівельних, машин різного призначення; працює більше десятка розвантажувальних машин і машин для укладання бетону на греблі; протяжність автомобільних трас складає декілька десятків кілометрів при достатньо великій завантаженості транспортом, який створює перешкоди при перевезенні вантажів.

Створення подібної системи вимагає розв'язування наступних задач:

1) вибір траси для підвозу бетону і конструкцій до греблі, а також щебеню й піску до бетонних заводів (при цьому можливі випадки використання загальнодержавних доріг, прокладання нових, прокладання спеціальних під'їзних шляхів та тунелів і т. д.);

2) вибір кількісного та якісного складу управління механізації;

3) вибір типів залізобетонних заводів, режимів їх роботи, місць розміщення;

4) вибір складу підйомно-транспортних, розвантажувальних і укладальних машин, режимів їх роботи;

5) вибір системи машин для виготовлення щебеню тощо.

Ясно, що в цьому випадку натурні експерименти можливі лише в досить обмеженому обсязі. Враховуючи унікальність такого будівництва, результати експериментів, що отримані в одних умовах, будуть малокорисними в інших.

Наведений приклад дає деяке уявлення про характер одного з типів складної виробничо-організаційної технічної системи.

Розглянемо більш детально основні характерні риси, які притаманні цій системі.

Навіть при поверховому розгляді цієї складної технічної системи нам довелось увести поняття підсистеми як деякої достатньо автономної частини всієї системи. Розчленування складної системи на підсистеми, як правило, здійснюється з певним елементом довільності й залежить як від прийнятих технічних рішень, цілей створення системи, так і від поглядів дослідника на систему. Це, в першу чергу, пояснюється різнорідністю елементів, що становлять складну систему. В розглянутому прикладі "транспортна" підсистема, яка відображає рух транспортних засобів по трасі від залізобетонних заводів до місця будівництва, може бути, в свою чергу, розчленована на більш дрібні підсистеми, наприклад, підсистеми перевезення бетону бетонозмішувачами та перевезення спеціальними транспортними засобами арматурних і залізобетонних конструкцій. Ці підсистеми, в свою чергу, можуть бути розподілені ще на більш дрібні підсистеми — це підсистеми організації перевезень, технічного обслуговування й ремонту транспортних засобів.

Як у цій виробничо-організаційній системі, так і в інших складних технічних системах одну з основних рис складає взаємодія виділених підсистем. Ця взаємодія виникає в результаті того чи іншого поділу системи на підсистеми. Як уже відзначалось, такий поділ системи на підсистеми має певний елемент довільності. Щоб забезпечити при цьому функціонування всієї системи як єдиного цілого, доводиться тим або іншим способом враховувати результати взаємодії однієї системи з іншою. Звичайно, така взаємодія зводиться до обміну сигналами між підсистемами, який здійснюється по каналах зв'язку, що прокладаються від однієї підсистеми до іншої. Ці канали зв'язку можуть як відповідати загальним каналам, що існують у системі, так і породжуватись прийнятий поділом системи на

підсистеми. Крім того, взаємодія здійснюється між зовнішнім середовищем і виділеними елементами системи. Врахування цієї взаємодії аналогічне врахуванню взаємодії між окремими підсистемами, тобто зовнішнє середовище представляється у вигляді такої підсистеми.

Таким чином, розгляд виробничої організаційної системи будівництва ГЕС показує, що складна технічна система представляється у вигляді багаторівневої конструкції із елементів, що взаємодіють між собою та із зовнішнім середовищем. Тут до елементів 1-го рівня належать підсистеми, які на початку розчленовані як системи, до елементів 2-го рівня - підсистеми, що утворюються при розчленуванні підсистем 1-го рівня і так до тих пір, поки утворені елементи не стануть простими для дослідження або керування.

Виробничо-організаційна система будівництва греблі ГЕС представлена у вигляді дворівневої конструкції із взаємодіючих моментів (рис. 2.1). На цьому рисунку стрілками показані не справжні канали зв'язку, а канали, що служать для врахування взаємодії, яка існує між виділеними підсистемами. Фактично обмін каналами відповідає в даному випадку, наприклад, входу або вигляду одиниці автотранспортного засобу з підсистеми, що розглядається. При цьому самі підсистеми виділені , виходячи з реально існуючих особливостей будівництва. Для спрощення прийнято, що працює два бетонних заводи, існує три місця укладання бетону з відповідними механізмами. В дійсності цих елементів може бути значно більше. Взаємодія із зовнішнім середовищем у даному випадку може звестися до відмови роботи заводів і укладальних машин, зміни дорожніх умов, виходячи із метеорологічних умов або інтенсивного руху стороннього транспорту тощо. У розглянутій системі підсистеми 1.2.1, 1.2.2, 1.4.1-1.4.3 являють собою не тільки бетонні заводи та розвантажувальні пристрої відповідно, але фактично включають в себе й транспортні засоби — як ті, що очікують завантаження (відповідно розвантаження), так і ті, що завантажуються (розвантажуються) в даний момент. Аналогічно підсистеми 1.3.1-1.3.k включають у себе не тільки параметри відповідних ділянок дороги, але й характеристики транспортних

засобів, що по них рухаються. Таке представлення складної системи у вигляді багаторівневої конструкції із взаємодіючих елементів має за мету вивчення системи за окремими частинами.

2.3. Технічна система «середовище-машина»

При розгляді виробничо-організаційної технічної системи будівництва греблі ГЕС ми бачили, що при виконанні тих або інших робіт спостерігається зв'язок між машиною і середовищем, з яким вона взаємодіє. Так, при виготовленні щебеню в кар'єрі проводиться ряд робіт: вскришні роботи, які виконуються з метою усунення пустої породи; відділення корисного шару від гірничого масиву і попереднє його розрихлювання для підготовки до навантажувальних робіт; транспортування гірничої маси до місця переробки; подрібнення та сортування кам'яного матеріалу; навантаження матеріалу в транспортні засоби й відправка його на залізобетонні заводи. Всі ці операції виконуються за допомогою спеціальних технологічних і транспортних машин. Цей приклад показує, що машини створюються для реалізації тих або інших технологічних або транспортних операцій, які здійснюються з метою зміни властивостей, форм, розмірів та місця розташування сировини для отримання готової продукції.

У даному випадку під середовищем розуміємо процес обробки і транспортування сировини, який нероздільний з тими або іншими машинами. Таким чином, взаємодія машини з середовищем становить технічну систему, підсистемами якої виступають середовище й машина, а характер їх взаємодії визначається структурою зв'язків між середовищем та машиною.

Комплексний розгляд машин і середовищ у вигляді технічної системи дає змогу виявити єдині закономірності взаємодії між ними. В процесі взаємодії із зовнішнім середовищем функціонування машини залежно від принципу дії проходить неперервно, дискретно або циклічно. Функціонування машини при кожному виді взаємодії проходить за своїми

внутрішніми циклами. Ці цикли характеризуються фазами зростання тих чи інших характеристик, їх стабілізації й спадання.

Безпосередніми середовищами функціонування (робочими середовищами), де здійснюється основна перетворювальна функція машини, є технологічне, організаційне або природне середовище. В них на машини здійснюють вплив сили опору, які можуть мати як корисний (відділення матеріалу від масиву, дроблення його тощо), так і негативний зношування елементів машини, вібрації, шум і т. д.) вплив. Статичні характеристики зміни сил дії машини на середовище досить часто мають вид експонет або парабол. Це дає можливість передбачити деяку загальну закономірність функціонування машини в певному середовищі при розгляді взаємодії машини з середовищем як системи.

Для забезпечення взаємодії між середовищем і машиною мають бути створені всі необхідні зв'язки із середовищем, забезпечено цілеспрямоване функціонування машини, тобто повинна бути побудована технічна система «середовище-машина». Для побудови такої системи, її аналізу й оптимального керування необхідна її модель (математична або фізична). Найбільш досконалими моделями таких систем слід визнати математичні моделі. Математичне моделювання системи «середовище-машина» може здійснюватись різними методами [9]. Взаємодія машини із середовищем, яка характеризується вихідними характеристиками машини $y_1 \ldots y_m$, полягає в тому, що ці характеристики неперервно зіставляються з характеристиками $z_1 ,\ldots,z_m$, які визначають зв'язки з тими або іншими елементами середовища. Метою цієї технічної системи є така поведінка системи, щоб розбіжності $A_1 ,\ldots,A_m$, між вихідними характеристиками машини (робочого органу) $y_1 \ldots y_m$ *і* середовища $z_1 ,\ldots,z_m$ постійно були рівні нулю, які б нові елементи середовища не вступали в контакт з машиною. Якщо = Z1,...,y_m =z_m, то машина адекватно відображає стан середовища і мета системи «машина-середовище» вважається досягнутою. Якщо ж мають місце розбіжності, не рівні нулю, то через механізм зворотного зв'язку здійснюється дія через

приводні механізми на машину і через робочі органи машини на середовище. Гнучкість та мобільність системи визначається кількістю вхідних функцій машини $и_i$, якими можна подіяти на систему. Чим більше функцій $и_i$, тим система більш гнучка і легше пристосовується до зміни середовища. З іншого боку, кількість довільних вхідних функцій $и_i$ залежить від кількості накладених в'язів на систему. Ця залежність при збільшенні кількості s спочатку росте, а потім спадає. Кількість вхідних функцій машини може бути зведена до нуля. В цьому випадку будемо мати систему з жорсткою структурою, яка при найменшій зміні середовища не може правильно функціонувати.

Які властивості повинна мати система, щоб забезпечити правильне функціонування? По-перше, вона повинна мати таку структуру (будову), яка дає можливість у процесі функціонування знімати частину накладених в'язів. По-друге, збільшення машиною кількості вихідних характеристик (координат робочого органу) підвищує рівень функціонування системи «машина-середовище». По-третє, система може мати можливість зміни структури на базі перерозподілу функцій, призначень рівнів керування, координуючих й погоджуючих елементів і організовувати самоорганізацію взаємодій між машиною та середовищем.

Відзначимо, що зменшити розбіжності $A1,..., A_m$ можна шляхом дії як на машину, так і на середовище, наприклад, за допомогою робочих органів машини.

Максимальна ефективність системи «машина-середовище» буде забезпечена при повному врахуванні взаємодії робочих органів з середовищем і виборі такої будови машини, яка б запезпечила оптимальну взаємодію. Це може бути досягнуто, коли машина і середовище розглядаються як єдина система, яка є підсистемою більш високого рівня системи — системи машин для виготовлення тієї чи іншої продукції.

2.4. Система машин

Одним з основних стратегічних напрямів матеріального виробництва в різних галузях є створення інтегральних виробничих комплексів (ІВК) і їх автоматизація. Побудова системи машин для забезпечення оптимального (найкращого) функціонування ІВК є однією з найважливіших проблем сучасного виробництва. Проблема створення системи машин охоплює значну сферу діяльності сучасної людини та включає в себе проблеми пов'язані зі створенням машин, дослідженням їх функціонування в різних середовищах, керуванням при виконанні .різного роду технологічних процесів. Проблеми створення й використання системи машин включають у себе першочергові завдання розробки нових машин і правильного системного їх використання при раціональних навантаженнях, погодженого функціонування й ефективної експлуатації.

На шляху створення системи машин необхідно розв'язувати ряд наукових проблем, де центральною є проблема взаємодій, які виникають між середовищем, машиною і людиною в світі сучасних технологій. Про взаємодію машини та середовища вже говорилося раніше, а роль людини в цих процесах є головною, бо заради неї створюються всі системи і процеси. В той же час людина в цьому бере безпосередню участь і тому її взаємодія з машинами повинна бути раціональною. Всі процеси, що виконуються більш продуктивно і ефективно машиною, повинні передаватись машині, а те, що краще виконує людина або чого не можуть виконати машини, повинно залишитися людиною. Світ сучасних технологій досягає такого рівня що за людиною залишаються тільки окремі елементи технологічних процесів, а все інше виконують машини. Прикладом тут може бути електронне машинобудування, де практично всі технологічні і транспортні операції механізовано й автоматизовано. За людиною залишаються тільки деякі функції керування, і то лише ті, які краще виконує людина. На жаль, цього не можна поки що сказати про галузь будівельного виробництва, де значна

частина технологічних операцій, особливо оздоблювальних, виконується людиною або механізмами й машинами при безпосередній участі людини.

Створення системи машин становить розвиток системи "середовище-машина" на основі цілеспрямованої організації багаторівневої структури, яка включає в себе машини, сукупність технологій та системи керування ними. В такій структурі може бути досягнутий рівень самоорганізації, тобто утворення, в якому заданий стан системи встановлюється без постійно діючого зовнішнього організаційного впливу. Принципи організації і самоорганізації в системі машин можна здійснити за рахунок оптимального комплектування машин, упровадження виробів автоматизації, вдосконалення механізмів координації керування, організації виробництва, нововведень і багатьох інших факторів прогресивного розвитку сучасних технологій.

Система машин є складною технічною системою, для впорядкування й організації якої необхідні значні ресурси. Перетворення системи «середовище-машина» в систему машин необхідно здійснювати з використанням закономірностей системних диференціації та інтеграції. Системна диференціація — це використання розбіжностей і невідповідностей між складовими частинами системи. Якщо складові системи стають досить різними за своєю організованістю й досить по-різному реагують на зовнішні дії, то це може стати причиною розпаду і зникнення системи. Розпаду системи протидіє системна інтеграція (об'єднання), тобто зростання цілісності системи, зміцнення зв'язків і взаємозалежність та взаємовпорядкованість її складових частин.

Створення системи машин може базуватись на наступних основних концепціях. По-перше, системи машин є засобами інтенсифікації виробництва та інших сфер людської діяльності. Отже, функціонування системи машин повинно розглядатись як процес погодженої взаємодії техніки і персоналу в середовищі сучасних технологій.

По-друге, створення системи машин передбачає підвищення інтегральної ефективності виробництва, тому конкретне застосування машин

повинно бути погоджене з їх циклами життя (тривалістю служби), а також ураховувати фази функціонування середовищ. Економічна ефективність сучасного виробництва може бути забезпечена лише при високій технічній ефективності системи машин.

По-третє, система машин повинна утворювати гармонічну єдність енергетичних, інформаційних і конструктивних властивостей сучасних технічних систем.

По-четверте, система машин має бути запроектована таким чином, щоб вона могла функціонувати при дотриманні певного співвідношення між рівнем організованості середовища і конструктивно-функціональними можливостями (складністю) машин. Це приводить до можливості вироблення кардинально нового принципу у створенні машин і прогнозування процесу їх розвитку. Цей принцип полягає в тому, що прогрес машинобудування повинен визначатись кількісними співвідношеннями між вихідними параметрами машини та їх взаємодією з середовищем.

Прикладом системи машин може бути комплекс машин, який використовується для дроблення і сортування кам'яного матеріалу з метою отримання щебеню й гравію [11]. Для одержання щебеню шляхом дроблення твердих порід каменю і для виділення гравію з гравійно-піщаної суміші створюють спеціальні комплекси — дробильно-сортувальні заводи, де повністю механізовані та зв'язані між собою у визначеній послідовності основні операції обробки (дроблення, сортування, виділення сторонніх домішок) утворюють безперервний технологічний процес (рис. 2.2). За технологією камінь, що був доставлений на завод у вигляді крупних кусків, подається для дроблення за допомогою транспортера 1 у дробильну машину 2. Отриманий дроблений матеріал передається від дробильної машини 2 на грохот 3, де він розділяється на фракції за крупністю. У дробильному матеріалі майже завжди є надмірні куски, що потребують додаткового дроблення.

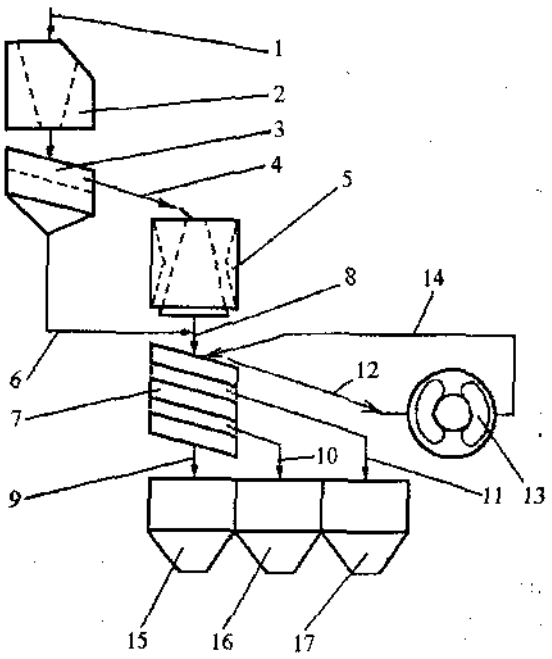

Рис. 2.2. Схема процесу дроблення кам'яних матеріалів

При сортуванні дробильного матеріалу на грохоті 3 такі надмірні куски затримуються на верхньому його ситі, звідки за допомогою транспортера 4 переміщуються в спеціально поставлену дробарку 5 повторного дроблення.

Операція сортування, що здійснюється на грохоті 3, виконується з метою відокремлення з продукту попереднього дроблення лише надмірних його кусків для передачі їх у дробарку 5 повторного дроблення. Решта матеріалу, що пройшов крізь отвори сита грохота 3, направляється 5а допомогою транспортера б для остаточного його сортування за крупністю на грохоті 7, куди одночасно по транспортеру 8 потрапляє з цією ж мстою і продукт повторного дроблення з дробарки 5. На грохоті 7 матеріал розділяється на три фракції й за допомогою транспортерів 9-11 направляється в приймальні бункери готової продукції. Крім того, на верхньому ситі грохоту 7 залишаються куски матеріалу, які направляються ше для одного циклу дроблення по транспортеру 12 у дробарку 13. Вихідний продукт цього дроблення відправляється за допомогою транспортера 14 на грохот 7 кінцевого сортування.

З бункерів І5-17 за допомогою дозаторів щебінь різних фракцій певними

порціями відправляється на залізобетонні заводи або для Інших потреб.

На рис. 2.2 показано схему одного з процесів дроблення кам'яних матеріалів. У той же час на практиці використовуються й Інші схеми таких технологічних процесів. Як бачимо з наведеного прикладу, для реалізації такого технологічного процесу необхідна

система машин. Залежно від потреб виробництва ці машини повинні бути узгоджені з позицій функціональних можливостей, продуктивності, надійності тощо. Крім того, на кожному з етапів процесу дроблення можуть бути використані машини різних типів. Так, дробарки 2, 5 і 13 можуть бути щоковими, конусними, валковими, ударними. Аналогічно різних типів можуть бути грохоти й транспортери. Кожний із типів цих машин має свої переваги і недоліки. Однак найвищу ефективність від роботи цих машин можна отримати тільки в тому випадку, коли вони будуть розглядатись як система машин для виконання повного технологічного процесу, а не як окремі машини. В той же час кожна з цих машин становить підсистему системи машин, тому розгляд окремої машини як системи в питаннях їх раціональної будови, міцності елементів, режимів руху, надійності, металомісткості є також дуже актуальною проблемою.

2.5. Машина як технічна система

Окрему машину можна розглядати як основний елемент системи машин, що використовуються для виконання того чи іншого робочого процесу. Поняття "машина" охоплює велику кількість найрізноманітніших об'єктів, які використовуються людиною для задоволення її потреб.

Згідно з визначенням І. І. Артоболевського, під "машиною" розуміють пристрій, який створено людиною в результаті вивчення та використання законів природи з метою полегшення або повної заміни її фізичної і розумової праці, підвищення продуктивності й точності при виконанні технологічних, транспортних та інших процесів. У більш короткій формі

поняття "машина" може бути визначене наступним чином: машина — це технічна система, яка виконує механічні рухи для перетворення енергії, матеріалів і інформації. Оскільки в машинах механічні рухи здійснюються за рахунок окремих елементів, які знаходяться у зв'язках між собою й об'єднані в деяку цілісність з певною метою, то вони мають усі ознаки систем.

Кожна машина як технічна система має певне призначення і будову (структуру). Призначення машини визначається функціями, які вона може виконувати в тих чи інших умовах використання. Для реалізації цих функцій створюється структура машини, яка визначає її властивості. Структурно машина як система може складатися з підсистем різних рівнів і окремих елементів, які утворюють ці підсистеми.

За функціональною ознакою машини можна розділити на енергетичні, робочі та інформаційні.

Енергетичною машиною називається машина, що використовуються для перетворення будь-якого виду енергії в механічну енергію або навпаки. В першому випадку вона носить назву машини-двигуна, а в іншому — машини-генератора. До машин-двигунів належать: електродвигуни, парові і гідравлічні турбіни, двигуни внутрішнього згоряння тощо. До машин-генераторів відносять електрогенератори, компресори, гідравлічні насоси.

Робочою машиною називається машина, яка використовується для перетворення зміни положення й дослідження властивостей матеріалів. Робочі машини поділяються на технологічні, транспортні і випробувальні.

Технологічною машиною називається робоча машина, в якій перетворення матеріалу проходить шляхом зміни властивостей, форми, розмірів і положення. До технологічних машин відносять металообробні верстати, будівельні, сільськогосподарські та інші машини. *Транспортні* машини призначені для переміщення в просторі різного роду об'єктів. До цих машин належать: тепловози, електровози, автомобілі, ванта-жопідйомні й транспортні машини, ліфти і т. п. *Випробувальні* машини використовуються для дослідження, наприклад, впливу механічної дії в заданих умовах на

властивості матеріалу або виробу (розривна машина, машина тертя, твердомір тощо).

Інформаційною називається машина для перетворення інформації. Основу цих машин складають електронно-обчислювальні машини (ЕОМ), які використовуються для виконання розрахунків, логіко-математичних операцій, моделювання процесів, розв'язування задач оптимізації і т .д.

Важливе місце в сучасній техніці займають машини-автомати, які використовують, головним чином, у тих випадках, коли необхідно виконувати одну і ту ж операцію багато разів. Прикладом машин-автоматів можуть бути металообробні верстати-автомати. Крім того, використовуються машини з технічними засобами на базі ЕОМ, які автоматично виконують складні процеси за попередньо створеною програмою в оптимальному режимі.

Для виконання функцій контролю і керування технологічними процесами та підприємствами використовують автоматизовані системи керування, які працюють при участі людини й системи автоматичного керування, наприклад, руху робота, які функціонують без будь-якої участі людини. Роботи являють собою маніпулятори з пристроями керування на базі ЕОМ, використовуються для здійснення рухів і функцій керування, що замінюють аналогічні функції людини.

Згідно з функціональною ознакою будівельні машини можна розділити на там класи: горизонтального безрейкового транспорту; вантажопідйомні і монтажні; неперервного транспорту; наватажувально-розвантажувальні; землерийні; бурові; пальові; механічної обробки (дроблення й сортування) кам'яних матеріалів; приготування, транспортування та укладання бетонних сумішей і розчинів; виробництва залізобетонних виробів; дорожні; оздоблювальні й механізований інструмент.

Кожний із цих класів машин поділяється на групи, які розрізняються характером робочого процесу (взаємодією з матеріалом) або режимом роботи. Так, землерийні машини поділяються на землерийні та землерийно-

транспортні. В свою чергу, землерийні машини можуть бути циклічної дії (одноковшові екскаватори) й неперервної дії (багатоковшові або роторні екскаватори).

Крім того, групи можуть поділятися на типи, які відрізняються конструкціями окремих елементів. Усі типи машин мають ряд типорозмірів, які відрізняються між собою величинами основних технологічних параметрів (вантажопідйомність, виліт та висота підйому вантажу — для крана, місткість ковша — для екскаватора), але мають в основному близьку будову.

У функціонально-структурному плані сучасні робочі машини складаються з ряду взаємодіючих частин (елементів): енергетична частина, передавальні механізми, виконавчі (робочі) органи і комплекс вимірювальних пристроїв та пристроїв контролю, регулювання, керування (рис. 2.3).

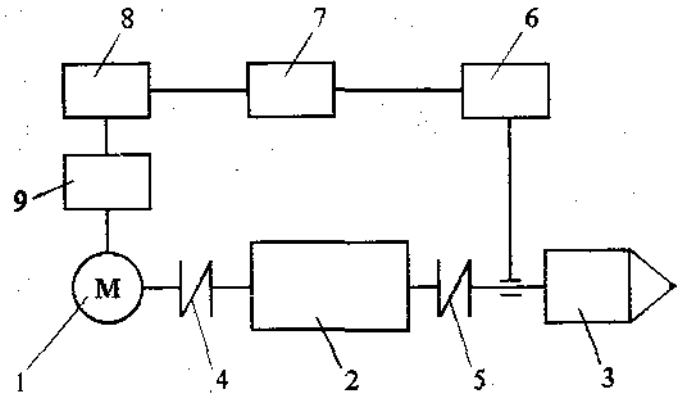

Рис. 2.3. Функціонально-структурна схема будови машини: 1 - двигун; 2 - передавальний механізм; 3 - робочий орган; 4, 5 - елементи зв'язку між елементами машини; б, 7, 8, 9 - відповідно вимірюючі, контролюючі, керуючі та регулюючі пристрої

Виконуючі (робочі) органи — це головна частина машини. Вони виконують комплекс технологічних операцій по перетворенню матеріалу (предмета праці) в продукт виробництва. Оскільки робочі органи перетворюють різні природні матеріали, процеси і сировину в продукт виробництва, то й принцип дії, конструктивні форми та інші характеристики

робочого органу визначаються властивостями матеріалу для обробки, параметрами і особливостями процесу, який реалізує робочий орган. Наприклад, робочий орган екскаватора (ковш) пристосований для копання і виймання грунту, дробильні плити щокової дробарки - до подрібнення вхідної будівельної сировини.

Енергетична частина (двигун) забезпечує енергією механізми і робочий орі-ан шляхом перетворення різних видів енергії в механічну енергію. Залежно від необхідних зусиль на робочому органі та режиму його руху двигун має ті чи інші механічні характеристики (залежності швидкості руху від зусилля). Двигуни пройшли в своєму розвитку ряд історичних етапів, починаючи від винаходу парової машини до широкого використання електродвигунів. Причому електродвигуни застосовуються для кожного функціонального механізму. На сьогодні в транспортних, будівельних та інших машинах широко застосовуються двигуни внутрішнього згоряння, які в майбутньому, певно, матимуть обмежене застосування, бо мають значну токсичність і спричиняють небажаний екологічний вплив, а також через обмеженість вуглецеводневого палива.

Передавальний механізм становить проміжну ланку між двигуном та робочим органом, завдання якої полягає в передачі енергії руху від двигуна до робочого або виконавчого механізму. Енергія і рух двигуна перетворюються за допомогою різних передавальних механізмів (зубчастих, ланцюгових, пасових передач, валів, муфт, гідропередач) у необхідні рухи робочого органу. Оскільки вал двигуна має більшу швидкість обертання, ніж основний вал робочого органу, то завданням передаточного механізму є зменшення швидкості обертання вала двигуна до рівня швидкості основного вала робочого органу. Передавальний механізм з одного боку обумовлений принципом дії, характером руху та іншими параметрами робочого органу, а з іншого — визначається типом і конструкцією двигуна. Передавальний механізм машини зв'язує в єдине ціле

технічний пристрій, який виконує задані технологічні функції у ви-

робничому процесі.

Контрольно-керуюча частина машини забезпечує автономне функціонування технічного пристрою, який виконує той чи Інший робочий процес шляхом самовлаштування, саморегулювання і самоконтролю. В цьому процесі людина стає складовою виробничого процесу, замість того, щоб бути його головним учасником і переключитися на творчі види діяльності.

Аналіз фумкціонально-структурної будови машини показує, що машина складається з елементів (підсистем), які знаходяться у взаємозв'язках між собою. Крім того, кожна з розглянутих підсистем розділяється на підсистеми більш низького рівня. Так, двигун внутрішнього згоряння включає в себе підсистеми: циліндро-поршневої групи, перетворення поступального руху поршня в обертальний рух колінчастого вала, живлення, газорозподілення, запалення, мастильна, охолодження. Причому між цими підсистемами двигуна існує певний взаємозв'язок. Аналогічні підсистеми більш низького рівня можна видалити в передаточному та виконавчому механізмі, вимірювальному, контролюючому і регулюючому пристроях. Усе це показує, що машина становить складну технічну систему з ієрархічною структурною будовою.

Крім того, життєвий цикл машини становить складну систему. Структура життєвого циклу машини включає в себе процеси її розробки, виготовлення, експлуатації й ремонту, які охоплюють час від виникнення ідеї створення машини до зняття ЇЇ з експлуатації.

Життєвий цикл, як правило, включає в себе наступні основні стадії або форми (рис. 2.4):

1. Формування вимог до машини і розробка технічного завдання (ТЗ).

2. Проектування машини.

3. Виготовлення, випробовування та доведення дослідного зразка машини.

4. Серійне виробництво машини.

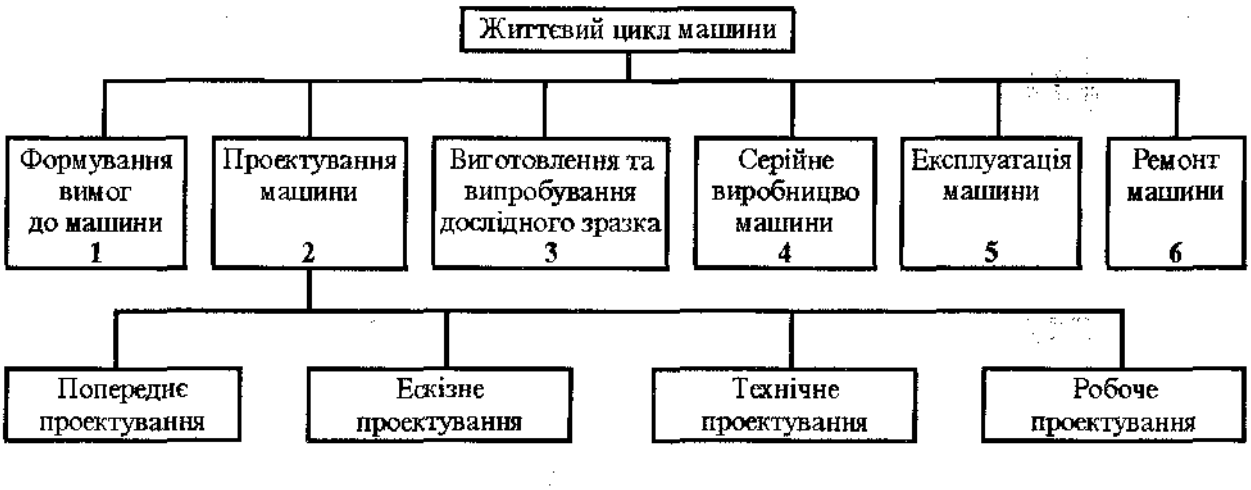

Рис. 2.4. Стадії життєвого циклу машини

5. Експлуатація й цільове застосування машини.

6. Ремонт машини.

Перша стадія життєвого циклу машини становить зовнішнє проектування. На цій стадії розв'язуються питання, пов'язані із з'ясуванням цілей, заради досягнення яких створюється машина. Уточнюється коло завдань, які планується розв'язувати за допомогою машини, а також досліджуються властивості зовнішнього середовища і вивчаються характеристики його впливу на машину. Результатом зовнішнього проектування є технічне завдання на розроблення проекту, яке містить основні вимоги до машини й взаємодії її із зовнішнім середовищем, що дає можливість розв'язувати поставлені задачі по створенню машини.

Другу стадію, щоб підкреслити її відмінність від першої, часто називають внутрішнім проектуванням. Це стадія безпосередньої розробки проекту машини. Тут розв'язуються питання, пов'язані з визначенням внутрішньої структури машини, з технічними рішеннями її підсистем і конструкцій їх елементів, а також параметрів та режимів експлуатації машини. Мета внутрішнього проектування полягає в розробленні всієї необхідної проектно-конструкторської документації, яка складає робочий проект машини, що задовольняє вимоги технічного завдання, тобто вимоги зовнішнього проектування.

З інформаційної точки зору внутрішнє проектування є процесом перетворення вхідної інформації про об'єкт проектування, про стан знань у даній галузі, про досвід проектування об'єктів аналогічного призначення у вихідну інформацію у вигляді проектно-конструкторської і технологічної документації, яка виконана в певній формі й містить опис об'єкта для його матеріальної реалізації, тобто виробництва. З точки зору теорії прийняття рішень проектування становить процес прийняття проектно-конструкторських рішень, котрий направлений на отримання опису машини із заданим ступенем деталізації, що задовольняє вимоги технічного завдання.

Друга стадія (проектування) пов'язує наукові дослідження з практичною реалізацією і становить процес розроблення опису, достатнього для створення ще не існуючої машини. Ця стадія, без сумніву, є однією з найважливіших стадій у життєвому циклі машини. В ній можна виділити чотири головних етапи:

1) попереднє проектування, або етап технічних пропозицій;

2) ескізне проектування; 3} технічне проектування; 4) робоче проектування.

На першому етапі здійснюється формування технічної концепції і основних (облікових) параметрів машини, що забезпечують виконання вимог технічного завдання. На цьому етапі усуваються некоректності у вимогах технічного завдання й проходить погодження вимог зовнішнього проектування з можливостями внутрішнього проектування. Характерним для цього попереднього проектування є відсутність детальної структуризації. Тут приймаються рішення з питань, які відносяться до машини в цілому. Виходом цього етапу є технічні пропозиції на розроблення проекта машини.

Основною метою ескізного проектування є уточнення характеристик та параметрів машини, що пов'язані з проектно-конструкторською розробкою її основних підсистем і агрегатів та формування їх обліку. Останнє здійснюється на основі ієрархічної (рівневої) структуризації машини й супроводжується проведенням широкого комплексу розрахунків і

експериментальних досліджень. Проробка повинна мати таку глибину, щоб можна було зробити порівняльний аналіз показників якості проектних рішень різних варіантів машини з урахуванням конструктивних та експлуатаційних характеристик. Виходом цього етапу є ескізний проект машини.

Етап технічного проектування передбачає розроблення комплексу конструкторських документів, що містять остаточні технічні рішення машини. В технічному проекті оцінюють остаточну відповідність проектних рішень вимогам технічного завдання і ступінь складності виготовлення машини, а також наводять правила її експлуатації та ремонту. Виходом цього етапу є технічний проект, який є основою для розроблення робочої конструкторської документації.

Накінець, на "етапі робочого проектування (конструювання) глибина проробки проекту досягає рівня деталей. Результат роботи на цьому етапі — робочий проект, який становить комплект конструкторської документації на машину, інструкцій до виготовлення і складання її елементів у вузли й агрегати та машини в цілому, а також інструкцій по випробуванню, експлуатації і ремонту машини.

На стадіях проектування проходять процеси постановки й розв'язування задач проектування, що задовольняють задані умови. Від цих стадій значною мірою залежать наступні стадії життєвого циклу машини.

На третій стадії життєвого циклу машини здійснюється виготовлення і випробування дослідного зразка машини. Дослідний зразок машини може створюватися розробником, тобто організацією, що Розробляє проектну документацію, або виробником — організацією, що розробляє проектну документацію, або виробником — організацією, що здійснюватиме серійне виробництво машин. На цьому етапі може виготовлятись окремий зразок або дослідна партія машин. Під час виготовлення дослідного зразка відпрацьовується технологія виготовлення окремих деталей і складання їх в окремі вузли, агрегати та машини в цілому. Після виготовлення дослідного зразка машини проводять попередні заводські випробування, під час яких

відпрацьовують раціональні експлуатаційні режими, виявляють необхідність якихось змін у конструкції машини чи її складових частинах. За результатами попередніх випробувань, які проводять разом розроблювач і виробник, вирішують питання про можливість здійснення приймальних випробувань, коригують конструкторську документацію, вносять зміни в конструкцію дослідного зразка. Після успішних попередніх випробувань вирішується питання про проведення приймальних випробувань, які залежно від характеру зв'язку між співвиконавцями можуть бути відомчими, міжвідомчими і державними. Приймальні випробування засвідчують відповідність машини розробленій технічній документації і можливість запуску машини у серійне виробництво.

Четверта стадія — це стадія серійного виробництва машини. Здійснення цієї стадії починається з технологічної підготовки виробництва, яка направлена на розроблення технологічних ліній виготовлення деталей і складання їх у вузли, агрегати і машину в цілому. Тут також здійснюється підбір обладнання, матеріалів, інструментів тощо. На цій стадії повинні бути задоволені технологічні вимоги до процесу виготовлення машини, які направлені на зменшення собівартості виробництва, зменшення витрат енергії і матеріалів, а також забезпечення функціональних показників, соціальних, експлуатаційних і економічних вимог. Це може бути здійснено шляхом вибору найбільш ефективних способів виготовлення деталей і зміцнення їх поверхневого шару, а також підвищення чистоти і точності обробки деталей. Результатом цього етапу ϵ серійний випуск машин тієї чи іншої якості, які поступають в експлуатацію.

На п'ятій стадії (стадії експлуатації) здійснюється використання машини. На цій стадії проявляються конструктивні, технологічні та інші властивості машини. Основна вимога до машин в процесі експлуатації полягає у виконанні своїх функцій за призначенням з мінімальними експлуатаційними витратами. Останні в значній мірі залежать від рівня надійності машини, яка, в свою чергу, залежить від того, наскільки правильно вирішені при

проектуванні питання міцності, жорсткості, зносостійкості і точності. Крім того, експлуатаційна надійність машини значно залежить *від* умов експлуатації і режимів руху приводних механізмів, які визначаються якістю керування машиною.

Забезпечення необхідної експлуатаційної надійності машини залежить від її технічного стану, який повинен підтримуватись на необхідному рівні шляхом своєчасного проведення технічних обслуговувань і ремонтів. Якість проведення останніх в значній мірі залежить від рівня ремонтопридатності машини, яка забезпечується тоді, коли в конструкції машини передбачена система контролепридатності, доступності, легкозйомності, взаємозамінності, стандартизації і уніфікації. Рівень технічного стану машини визначається шляхом діагностування окремих елементів, підсистем і машини в цілому як системи.

Всі розглянуті стадії життєвого циклу машини супроводжуються науковими дослідженнями, тобто на кожній стадії виникають задачі, які не можуть бути розв'язані традиційними способами, а вимагають розроблення спеціальних методів. Кінцева мета цих досліджень полягає в розробленні таких машин, які б мали високу економічну ефективність. У процесі проектування необхідно враховувати значну кількість факторів, які визначають ефективність виробництва і використання машин. Інколи спрощення конструкції машини й технології її виготовлення приводять до значних витрат на обслуговування та ремонт у процесі експлуатації. Глибокі економічні дослідження показують, що головними критеріями економічності машини досить часто є не ЇЇ вартість, яка визначається витратами на матеріали, дослідження, проектні роботи і виготовлення машини, а ЇЇ продуктивність, надійність, витрати енергії, вартість обслуговування, ремонтів тощо.

З аналізу етапів життєвого циклу машини можна зробити висновок, що вони знаходяться в тісному взаємозв'язку між собою. Зовнішнє проектування визначає призначення машини та вимоги до неї. На основі зовнішнього

проектування проводиться внутрішнє проектування, яке визначає структуру машини, конструкції і компоновку (розміщення) ЇЇ складових частин. Прийняті проектні рішення визначають технологію виготовлення деталей машини й складання машини. Конструкція і рівень технології виготовлення машини суттєво впливають на її експлуатаційну надійність. Умови експлуатації визначають рівень технічного стану машини. При досягненні певного технічного стану машини проводяться ті чи інші види ремонту з деякою періодичністю.

Отже, все наведене показує, що життєвий цикл машини можна вважати складною технічною системою, підсистемами якої є окремі

стадії створення машин, які, в свою чергу, діляться на підсистеми більш низьких рівнів. Так, процес проектування поділяється на стадії зовнішнього і внутрішнього проектування, в той же час останнє ділиться на попереднє, ескізне, технічне й робоче. Тому до процесу створення машини необхідно підходити як до створення складної технічної системи з урахуванням взаємозв'язків між окремими елементами. Неврахування цього може привести до небажаних результатів у процесі створення машин.

В історії створення машин є немало суттєвих прорахунків у цьому напрямі. Можна навести загальновідомий приклад з практики створення літаків-винищувачів [12]. Досвід другої світової війни показав, що далеко не всі прогнози підтверджуються практикою боїв. Тактична доктрина кінця 30-х років виходила з того, що повітряна війна буде проходити на великих висотах. У всіх країнах світу намагались підняти стелю польоту бойових літаків. МІГ-3 був найбільш яскравим проявом цієї доктрини... Але в перші ж місяці війни пересвідчилися, цю німецькі льотчики на літаках-винищувачах "Мессершмstт", які мали меншу висоту підйому, ніж МІГи, не ведуть боїв на тих висотах, де вони слабкіші від МІГів. Навпаки, вони старались зав'язати всі бої на малій висоті, де більш важкий МІГ програвав у маневрі. До того ж тривалість польоту МІГа на малих висотах виявилася недостатньою. Коли всі ці обставини стали очевидними, конструктори МІГа постаралися зменшити

вагу літака за рахунок зняття частини зброї І деяких інших рішень. Але це не допомогло, і в результаті було прийняте рішення про припинення виготовлення літаків МІГ-3. Ці приклади показують, що принципова помилка, яка була допущена при зовнішньому проектуванні {не було враховано всі фактори при виборі параметрів літака), незважаючи на витрати значних зусиль і коштів, кінець кінцем привела до невдачі.

Кінцева мета створення будь-якої машини полягає в ефективному її функціонуванні, яке залежить не тільки від досконалості конструкції, але й від способів ЇЇ використання і від режимів керування нею. Особливо це відноситься до швидкісних машин циклічної дії, коли доводиться чергувати часті пуски та зупинки. Тому необхідно вибирати такі режими функціонування машин, при яких для тієї чи іншої конструкції буде досягнута найвища ефективність. Усе це ще раз показує, що процес створення і використання машини необхідно розглядати як систему, в якій ураховуються зовнішні характеристики, конструктивні особливості її будови, режими функціонування тощо.

2.6. Задачі теорії технічних систем

Як було показано раніше, перш за все частинами складної системи є підсистеми різних рівнів, на які розчленовується початкова система. Оскільки підсистеми верхніх рівнів самі досить часто є складними і розчленовуються далі, то в дійсності вивченню підлягають найдрібніші, неподільні системи, які умовно називають елементами. Самостійною частиною є так звана схема спряження елементів, тобто схема, яка реалізує адресацію сигналів, що відображають взаємодію елементів. Визначено, що системи не називаються складними, якщо вони вивчаються як єдине ціле, без вказаної структуризації. В цьому І полягає основна різниця між простими і складними системами. Відміна між ними породжується не стільки самими дослідженнями як такими, а в основному задачами, які стоять перед

дослідниками.

Розглянемо більш детально складові частини складних технічних систем з метою вивчення завдань, які виникають при їх дослідженні. Звернемось спочатку до елементів технічних систем. Кожний елемент становить динамічну систему, яка функціонує в часі, з часом змінює свій стан під дією внутрішніх І зовнішніх причин, сприймає вхідні й видає вихідні сигнали в процесі взаємодії з іншими елементами системи. Перераховані властивості є загальними для різних елементів, однак кожний конкретній елемент має свої особливості.

У технічній системі будівництва греблі ГЕС можна виділити різнорідні елементи — залізобетонні заводи, розвантажувальні пристрої, систему машин для виготовлення щебеню і т. ін.

На роботу кожного елемента і системи в цілому досить суттєво впливають випадкові фактори (відмови технологічних машин і транспортних засобів, випадкові потоки транспорту на різних ділянках дороги від заводів до місця будівництва, випадкові тривалості виконання різних операцій). Усе сказане приводить до висновку про те, що для опису елементів складних технічних систем необхідно мати набір формальних математичних моделей, які задовольняють, з одного боку, загальним властивостям всіх динамічних елементів, а з іншого — враховують різні окремі особливості функціонування кожного конкретного елемента.

Таким чином, однією з найважливіших задач теорії технічних систем є пошук математичних моделей, які достатньо адекватно відображають специфіку роботи елементів складних систем. У той же час такі математичні моделі повинні допускати дослідження систем аналітичними або машинними (за допомогою ЕОМ) методами.

Далі буде розглянуто ряд таких моделей, які знайшли широке застосування в теорії складних технічних систем.

Основу функціонування складних технічних систем становить взаємодія елементів, яка представляється у вигляді механізму обміну сигналами. Цей

механізм включає в себе процеси формування вихідних сигналів та реагування на вхідні сигнали різними елементами, адресації сигналів і їх проходження по каналах зв'язку. Облік процесів формування вихідних і реагування на надходження вхідних сигналів належить до проблеми побудови елементів як динамічних технічних систем. Далі зручно вважати, що під час проходження по каналу зв'язку сигнал не підлягає викривленню, а сам процес передачі проходить миттєво. Такі канали називаються ідеальними. Вони виникають при розчленуванні системи у випадку, коли реальна система не має ніяких каналів зв'язку, а вони вводяться тільки для існуючої взаємодії елементів системи. Такі канали зв'язку існують у технічній системі будівництва греблі ГЕС. У реальній системі наявні реальні канали зв'язку (наприклад, система програмного керування рухом зварювального маніпулятора в арматурному цеху залізобетонного заводу), які, природно, е неідеальними. В цьому випадку їх зручно формалізувати як окремі елементи системи, які реалізують затримки і відхилення сигналів. Такі елементи з'єднуються з іншими елементами вже ідеальними каналами зв'язку. Отже, для вивчення процесу взаємодії елементів технічної системи достатньо розглянути схему спряження, яка реалізує адресацію сигналів у системі з ідеальними каналами зв'язку. При розгляді схеми спряження елементи представляються у вигляді багатополюсників, які мають певне число вхідних і вихідних контактів — кількість цих контактів різна для різних елементів. Якщо занумерувати всі вхідні контакти елементів система та її вихідні контакти, то задання схеми спряження означає зіставлення в кожній парі (i, j) (де i, j - порядкові номери відповідно вхідного й вихідного контактів) факту наявності або відсутності ідеального каналу зв'язку. Така конструкція враховує також взаємодію системи із зовнішнім середовищем — достатньо лише зовнішнє середовище уявити фіктивним елементом, який мас певну кількість вхідних контактів.

Зі схемою спряження елементів технічної системи виникають певні задачі. Перш за все, це задачі структурного аналізу складних технічних

систем, які виявляють різні відношення між елементами системи. Виникає інтерес до питання існування ланцюга каналів зв'язку, які з'єднують різні елементи. Більш глибоке дослідження передбачає врахування напрямку передачі сигналів, а також їх види. Під видом розуміють певну змістову інтерпретацію призначення сигналів, що передаються. Наприклад, деякі сигнали відповідають матеріальним потокам у системі, інші — інформаційним, треті служать цілям керування. Такий поділ сигналів, звичайно, проявляється у факті їх виникнення на певних вихідних клемах, що і дає можливість використати для вивчення вказаних задач структурні методи, які дають змогу знайти так звані типові структурні конфігурації (кола, цикли, контури тощо). Ці конфігурації відіграють важливу роль у визначенні можливостей системи з передачі і переробки сигналів. При цьому виявляються ще й такі властивості структур, як зв'язність, ієрархічність і т. ін.

Особливу роль відіграють формальні структурні перетворення, коли початкова структура системи перетворюється в іншу. Так, деяка підсистема може розчленуватися на ряд більш дрібних підсистем або, навпаки, ряд елементів об'єднується в одну підсистему. Такі перетворення відіграють важливу роль на етапі синтезу, коли розв'язується питання про можливість створення системи, яка володіє заданими властивостями і має деякий стандартний набір елементів. Разом з тим необхідно констатувати, що методи структурного аналізу й синтезу складних технічних систем розроблені значно слабше, ніж методи синтезу та аналізу динаміки окремих елементів[7, 8].

Виходячи з раніше сказаного, можна зробити висновок, що основним завданням теорії складних технічних систем необхідно вважати розроблення методів, які дають можливість на основі вивчення особливостей функціонування, отримання характеристик окремих елементів і аналізу механізму взаємодії між елементами отримувати характеристики системи в цілому. Єдиний метод, який існує на сьогодні і дозволяє знаходити характеристики системи в цілому, є метод моделювання систем. Причому

найбільш ефективними та продуктивними необхідно визнати методи машинного моделювання за допомогою ЕОМ. Однак метод моделювання може успішно застосовуватись лише при тій умові, що його реалізація повною мірою враховує особливості моделювання об'єктів, тобто моделювання має бути цілеспрямованим (у напрямі погодження модельних експериментів з властивостями системи, що моделюється).

Далі будуть розглянуті моделі, які є математичними і фізичними моделями елементів технічних систем. На основі моделей технічних систем можуть визначатися їх властивості й характеристики, здійснюватись аналіз їх функціонування та аналіз конструкцій, проводитись оптимізація структури систем та параметрів їх елементів, здійснюватися керування системами, в тому числі оптимальне, і т.д.

3. МОДЕЛЮВАННЯ ТЕХНІЧНИХ СИСТЕМ

3.1. Моделі і моделювання

Методи теорії технічних систем спираються на опис тих або інших фактів, явищ, процесів. Наші знання завжди відносні, тому будь-який опис на тій чи іншій мові також відображає лише деякі сторони явищ і ніколи не може бути абсолютно повним. Якщо користуватись мовою філософії, то можна сказати, що опис, відображаючи наші знання, завжди є відносним.

Останнім часом досить значне поширення отримало слово "модель". Поняття "модель" допускає багато різних тлумачень. Існує навіть класифікація моделей. У теорії систем під словом "модель", "модельний опис", розуміють деякий опис, який відображає саме ті особливості процесу, що вивчається, що цікавлять дослідника. Точність, якість такого опису визначаються насамперед відповідністю моделі тим вимогам, які пред'являються до дослідження, відповідністю отриманих за допомогою моделі результатів реальному ходові процесу.

Побудова моделей — завжди процедура неформальна І, звичайно, залежить від дослідника, його досвіду, таланту. Вона завжди спирається на деякий дослідний матеріал, у зв'язку з чим досить часто кажуть, що процес моделювання має феноменологічну основу. Модель повинна достатньо правильно відображати явища, однак одного цього ще мало. Вона має бути зручною для користування. Тому ступінь деталізації моделі, форма її представлення і т. д. визначаються метою дослідження й безпосередньо залежать від дослідника. Працюючи з одним і тим же дослідним матеріалом, різні дослідники можуть представляти його зовсім по-різному. Але при всій тій різниці існують загальні принципи моделювання, нехтувати якими недопустимо. Спинимося на цьому більш детально.

Основна задача аналізу технічних систем полягає у виділенні реальних рухів з множини уявно допустимих, сформулювати принципи їх відбору. Тут

і далі під рухом розуміємо більш широке поняття — зміна системи взагалі, всяка взаємодія її матеріальних об'єктів. Проблема моделювання полягає в описі цих принципів відбору в термінах і тих змінних, які відповідно до поглядів дослідника найбільш повно характеризують досліджувану систему. Принципи відбору звужують множину допустимих рухів, відкидаючи ті, які не можуть бути реалізовані. Чим більш досконала модель, тим вужчою стає множина реальних рухів, тим точнішим стає прогноз. У різних галузях знань принципи відбору рухів різні.

Сучасна наука розглядає три рівні систем: системи неживої матерії (до них належать технічні системи); системи живої матерії ("біологічні системи); системи, які пізнають себе (суспільні системи). Технічні системи відносяться до нижчого рівня систем, на якому основними принципами відбору є закони збереження: речовини, імпульсу, енергії тощо. Будь-яке моделювання повинно починатись Із вибору дослідником основних (або, як кажуть, фазових) змінних, за допомогою яких записують закони збереження. Але закони збереження не виділяють єдиного руху з множини уявно можливих і не вичерпують усіх принципів відбору. Необхідно ще враховувати другий закон термодинаміки (про необоротність процесів), принципи мінімуму дисипації енергії, стійкості руху й ін. Дуже важливе значення мають різного роду умови (обмеження): граничні, початкові і т. д.

На рівні біологічних і суспільних систем усі принципи відбору рухів, які справедливі для систем неживої матерії, зберігають свою силу. Той факт, що закони, які справедливі для систем неживої матерії, зберігають свою силу для систем живої матерії, довгий час був предметом дискусій. Особливо багато труднощів викликає другий закон термодинаміки. Це питання було розв'язане Л. фон Берте-Ланфі, котрий першим показав, що живі істоти являють собою відкриті системи. Це означає, що вони не можуть існувати без обміну речовиною й енергією з навколишнім середовищем (цим і пояснюється спостережуване у них зменшення ентропії). В живих системах, як і в неживих, процес моделювання починається із запису законів

збереження. Однак тут основні змінні мають інший вигляд, ніж у неживих системах.

У загальному випадку моделювання можна назвати особливою формою формалізованого підходу до дослідження складних систем. Теоретичною базою моделювання є теорія подібності.

Подібність — взаємно однозначна відповідність між двома об'єктами (системами), при якій відомі функції переходу від параметрів одного об'єкта до іншого, а математичні описи цих об'єктів можуть бути перетворені в тотожності. Теорія подібності дає можливість установити наявність подібності або дає змогу розробити спосіб її отримання.

Моделювання — це процес представлення об'єкта (технічної системи) адекватною (подібною) йому моделлю та проведення експериментів з моделлю для отримання інформації про об'єкт дослідження. При моделюванні модель виступає і як засіб, і як об'єкт досліджень, що знаходиться у відношеннях подібності до реальної інформації для керування системою, оцінити показники ц функціонування і тим самим на базі моделювання знайти найбільш ефективний варіант побудови та оптимальний режим функціонування реальної системи.

Відповідно до системного підходу в процесах створення й дослідження складних технічних систем моделювання їх елементів та функціональних підсистем виконується в декілька етапів і на різних рівнях залежно від ступеня деталізації системи. Методика моделювання безпосередньо залежить від рівня моделювання. Кожному рівню моделювання відповідає певне поняття системи, елемента системи, законів функціонування елементів системи в цілому і дії зовнішніх навантажень.

Так, при моделюванні механізму підйому вантажопідйомного крана на рівні елементів сам механізм виступає як система. Елементами цієї системи виступають складові механізму {рис. 3.1): двигун-1; гальмівний пристрій - 2; передавальний механізм - 3; барабанно-канатний механізм -4; поліспастова система - 5; захватний пристрій -6.

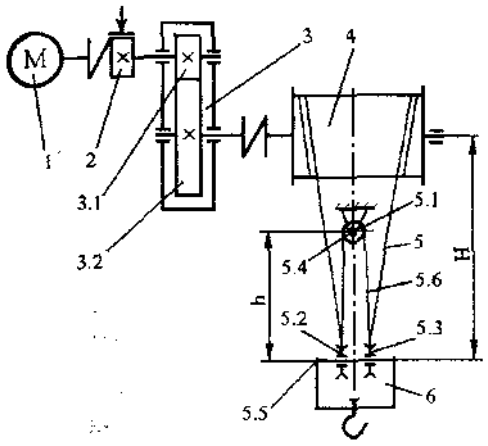

Рис. 3.1. Схема механізму підіймання вантажо-підйомного крана

Ці елементи зв'язані між собою у відповідності з функціональною схемою механізму підйому. Робота кожного елемента описується відповідною функцією, наприклад, передавальний механізм зменшує частоту обертання вала двигуна при передачі руху до барабана, а барабанно-канатний механізм перетворює обертальний рух барабана в поступальний рух захватного пристрою. Як зовнішні навантаження на цю систему можна розглядати вагу вантажу, вітрові навантаження і рушійний момент на валу двигуна. Розглянутий механізм являє собою динамічну систему, яка змінює свій стан у часі. Предметом дослідження цієї системи може бути визначення динамічних навантажень у канаті поліспасової системи як функції рушійного (гальмівного) моменту і параметрів механізму.

Той же механізм підіймання можна моделювати на рівні окремих елементів, наприклад, поліспастової системи. Тоді елементами такої системи можуть виступати блоки 5.1, 5.2, 5.3. осі 5.4 і 5.5, канат 5,6. Кожний із цих елементів виконує свої функції. Зовнішнім навантаженням для цієї системи може виступати динамічне навантаження у канаті 5.6, яке було визначено в моделі більш високого рівня.

Метою дослідження системи більш низького рівня може бути визначення динамічних навантажень, наприклад, на осі 5.4 І 5.5.

Залежно від ступеня деталізації опису складних технічних систем та їх

елементів можна виділити три основних рівні моделювання:

1. Рівень структурного або імітаційного моделювання складних систем із використанням їх алгоритмічних моделей (моделюючих алгоритмів) і застосування спеціалізованих мов моделювання, теорій множин, алгоритмів, графів, масового обслуговування, статистичного моделювання.

2. Рівень логічного моделювання функціональних схем елементів та вузлів складних систем, моделі яких представляються у вигляді рівнянь безпосередніх зв'язків (логічних рівнянь) і будуються із застосуванням апарату двозначної або багатозначної алгебри логіки.

3. Рівень кількісного моделювання принципових схем елементів складних систем, моделі яких становлять системи лінійних та нелінійних алгебраїчних, диференціальних або інтегро-диференціальних рівнянь, що досліджуються із застосуванням методів лінійної І нелінійної алгебри, методів функціонального аналізу, теорії ймовірності й математичної статистики.

Сукупність моделей технічної системи на структурному, логічному і кількісному рівнях моделювання являє собою ієрархічну систему, яка розкриває взаємозв'язок різних сторін опису технічної системи й забезпечує системний взаємозв'язок елементів та властивостей на всіх стадіях ЇЇ створення або дослідження. При переході на більш високий рівень абстрагування здійснюється згортання даних про систему, Що моделюється, а при переході до більш детального рівня опису — розгортання цих даних.

При створенні і дослідженні складних технічних систем застосовують методи аналітичного, чисельного, імітаційного, натурного й напівнатурного моделювання.

Аналітичні методи полягають у перетворенні символьної інформації, яка записана мовою математичного аналізу. При використанні аналітичних методів будується математична медаль системи, що описує її фізичні властивості за допомогою математичних співвідношень, наприклад, у вигляді диференціальних або інтегральних рівнянь. Моделі такого типу називають

аналітичними. Аналітична модель будується на основі понять, символіки і методів деякої теорії (наприклад, молекулярно-кінетичної теорії газів), яка визначає клас математичних аналогій, тобто основоположних математичних моделей. При аналітичному підході необхідні залежності виводяться з математичної моделі послідовним застосуванням математичних правил. Перешкодою при цьому може бути нерозв'язність рівнянь в аналітичному вигляді, відсутність первісної для підінтегральних функцій і т .д. Тому лише при певних властивостях моделі можна отримати розв'язок в явній аналітичній формі. Незважаючи на обмежені можливості аналітичного підходу, результати, які одержують в аналітичній формі, мають велику пізнавальну цінність та знаходять застосування при розв'язуванні широкого класу теоретичних і прикладних задач.

Останнім часом досить важливою стала задача впровадження в інженерну практику наближених аналітичних методів аналізу нелінійних об'єктів (процесів) різної фізичної природи в застосуванням _ ЮМ. Успішне розв'язування цієї задачі стало можливим завдяки розробці систем програмування, які використовуються для аналітичних і чисельно-аналітичних обчислень на ЕОМ. Завдяки створенню систем аналітичних обчислень та перетворень на ЮМ, яка дає змогу працювати безпосередньо з математичними формулами, застосування ЕОМ виявилось плідним для проведення складних розрахунків у небесній механіці, математичній фізиці тощо.

Чисельні методи базуються на побудові кінцевої послідовності дій над числами, яка приводить до отримання необхідних результатів. При наявності математичної моделі технічної системи застосування чисельних методів зводиться до заміни математичних операцій і співвідношень відповідними операціями над числами: заміна інтегралів сумами, похідних — відношеннями різниць, нескінченних сум — кінцевими і ТА У результаті цього будується алгоритм, який дає можливість точно або з допустимою похибкою обчислити значення необхідних величин на ЕОМ. Результатом застосування чисельних методів є таблиці, графіки, залежності, які

розкривають властивості системи. Чисельні методи по відношенню до аналітичних дозволяють розв'язувати значно більш широке коло задач, але при ньому отримані результати мають частковий характер і вимагають додаткової перевірки.

Характер процесів, які притаманні технічній системі й підлягають відображенню в моделі, може бути настільки складним, що побудова математичної моделі перетворюється в складну задачу, а аналіз моделі навіть чисельними методами може виявитися не результативним через трудомісткість або нестійкість алгоритмів у відношенні похибок апроксимації і округлень результатів обчислень. Відтворення в моделі динаміки складних просторово-часових відношень між елементами, що утворюють складну технічну систему, всієї багатогранності її зв'язків із зовнішнім середовищем, діючих у системі законів керування, адаптивних властивостей і рис цілеспрямованої поведінки практично неможливі чисто математичними засобами.

При дослідженні таких систем із використанням *методів імітаційного моделювання* широке застосування знаходять моделі, які являють собою змістовий опис технічних систем І форм алгоритмів. В описах відображаються як структура системи, що досягається шляхом ототожнення елементів системи з відповідними елементами алгоритмів, так і процеси функціонування системи в часі, які представляються в логіко-математичній формі. При цьому описи системи мають алгоритмічний характер, а самі моделі становлять програми для ЕОМ. Моделі такого типу називають *імітаційними* або алгоритмічними.

Особливість даного підходу до моделювання полягає в тому, що для побудови моделі використовуються алгоритмічні мови, які є більш гнучкими й доступними засобами опису складних систем, ніж мови математичних функціональних співвідношень. Завдяки цьому в імітаційних моделях складних технічних систем знаходять відображення багато деталей і функцій, що вимушено не враховуються в математичних моделях.

У теорії складних систем з притаманним їй імовірнісним підходом до моделювання систем широко використовується наближений чисельний метод аналізу імітаційних моделей — метод *статистичних* випробувань (метод Монте-Карло) [15]. Процес побудови та аналізу імітаційних моделей із застосуванням методу статистичних випробувань називається статистичним *моделюванням*. Статистичне моделювання є методом отримання за допомогою ЕОМ статистичних даних про процеси, які проходять в системі, що моделюється. Для отримання необхідних результатів статистичні дані опрацьовуються і класифікуються з використанням методів математичної статистики. Позитивна властивість статистичного моделювання — це універсальність, яка гарантує принципову можливість аналізу систем будь-якої складності з будь-яким ступенем деталізації досліджуваних явищ. Недолік статистичного моделювання — це трудомісткість процесу моделювання, тобто необхідність виконання великої кількості операцій над числами, і частковий характер результатів, який не розкриває залежності, а лише визначає її в окремих апріорно визначених точках. Імітаційні експерименти широко використовують у практиці створення й дослідження складних технічних систем, коли реальний експеримент неможливий.

Натурним моделюванням називають проведення досліджень на реальному об'єкті (системі) з наступною обробкою результатів експерименту на основі теорії подібності. При функціонуванні технічної системи відповідно до поставленої мети вдасться виявити закономірності проходження реального процесу. Необхідно відзначити, що такі різновидності натурного експерименту, як виробничий експеримент і комплексні випробування, мають високий ступінь достовірності. Методи натурного моделювання базуються на вимірюванні характеристик процесів, що проходять в реальних системах, й обробці результатів вимірювання з метою виявлення тих чи інших закономірностей, які цікавлять дослідника. Експериментальні дослідження дають найбільш достовірну інформацію, але вони носять частковий характер. Натурне моделювання може проводитися також на

фізичних моделях, які моделюють реальні процеси. Фізичні моделі будуються на основі теорії подібності.

Напівнатурне моделювання складних технічних систем здійснюється шляхом використання їх комбінованих моделей. У структуру таких моделей включають математичні співвідношення, які описують функціонування окремих елементів (підсистем), а також реальні елементи (підсистеми) або їх фізичні моделі, що є складовими елементами досліджуваної системи. В процесі дослідження комбінованих моделей може бути досягнута оптимальна взаємодія між обчислювальним і натурним експериментами. Методи напівнатурного моделювання ефективно використовують при дослідженнях складних технічних систем, які складаються з елементів різної фізичної природи. Ці методи враховують переваги математичного й натурного моделювання.

3.2. Фізичне моделювання

3.2.1. Основні поняття фізичного моделювання

У процесі проведення експериментальних досліджень можуть бути використані натурні об'єкти і їх моделі. В натурному експерименті засоби експериментального дослідження взаємодіють безпосередньо з об'єктом дослідження. В модельному експерименті дослідження проводять не з самим об'єктом, а з його замінником-моделлю. Модель тут відіграє подвійну роль. По-перше, вона являє собою об'єкт безпосереднього експериментального дослідження. По-друге, по відношенню до об'єкта, який вивчається, модель виступає як засіб експериментального дослідження.

В експериментальних дослідженнях досить широко застосовується фізичне моделювання технічних систем. Цим методом користуються при дослідженні складних явищ, коли неможливо побудувати задовільну математичну модель, або для перевірки адекватності математичної моделі.

Фізичне моделювання зберігає фізичну природу явищ, але змінює їх масштаб. Сенс фізичного моделювання полягає в тому, щоб за результатами дослідів на моделях можна було достовірно оцінити характер ефектів і кількісні взаємозв'язки між величинами, які визначають фізично подібні явища в натурних умовах.

Основою фізичного моделювання с теорія подібності, яка спирається на аналіз розмірностей. Об'єкти (явища, процеси, системи і т. ін.) є подібними, якщо у відповідні моменти часу у відповідних точках об'єктів значення змінних величин, що характеризують стан одного об'єкта (натури), пропорційні відповідним значенням величин іншого об'єкта (моделі). З цього визначення випливає, що в подібних об'єктах характеристики натурного об'єкта можуть бути отримані простим перерахунком із характеристик модельного об'єкта, що визначаються, як правило, експериментально. Для всіх величин певної розмірності таким множником є коефіцієнт подібності (множник масштабного перетворення).

До фізичного моделювання належать механічні, гідравлічні, електродинамічні, теплові та інші різновидності моделювання. Окремими вилами фізичної подібності є: геометрична подібність (подібність геометричних елементів фігур або тіл); кінематична подібність (подібність швидкостей для розглядуваних рухів); динамічна подібність (подібність систем діючих сил або силових полів різної фізичної природи — сили ваги, сили тиску тощо).

Механічна подібність включає в себе геометричну кінематичну й динамічну подібності. Електродинамічна подібність характеризується подібностями струмів, напруг, навантажень, потужностей, полів електромагнітних сил тощо.

Для подібних об'єктів загальними умовами є такі умови: модель і натурний об'єкт повинні бути геометрично подібними; діючі на модель навантаження повинні бути подібними навантаженням на натурні об'єкти; безрозмірні величини (коефіцієнт Пуассона, коефіцієнт тертя, відносна деформація тощо) для моделі та натурного об'єкта повинні бути однаковими;

матеріали моделі й натурного об'єкта можуть бути різними, але в досліджуваній галузі зв'язок напружень і деформацій має відповідати закону Гука.

3.2.2. Коефіцієнти і критерії подібності

При механічному моделюванні розрізняють просту та розширену подібність. При простій подібності коефіцієнти подібності для величин, що мають однакову розмірність (наприклад, геометричні розміри), повинні бути однаковими. При розширеній подібності вказані величини можуть мати різні коефіцієнти подібності.

Методика вивчення коефіцієнтів подібності в загальному випадку має такий вигляд. Наприклад, у механіці основними величинами вважають довжину *I,* час (і масу *m.* їх коефіцієнти подібності $v_l = l_н/l_м$, $v,= t_H/t_M$, $v_m= m_н/m_м$ вибираються .довільно, де $l_н$, $t_н$, $m_н$ — основні величини натурного об'єкта, $l_м$, t_M,,$m_м$ — відповідні величини моделі. В цих і подальших залежностях індекси "н" і "м" відносяться відповідно до параметрів натурного об'єкта й моделі.

Інші коефіцієнти подібності можуть бути отримані на основі фізичних законів. Для швидкості $V_H = l_н/t_н$ і $V_M = l_M/t_M$ коефіцієнт подібності $V_v = V_H/V_M = l_н t_M/l_M t_н$ можна визначити через коефіцієнти подібності довжини v_l і часу v_t у вигляді $v_v = v_l/v_t$. Відповідно до другого закону Ньютона сила F зв'язана з прискоренням w співвідношенням $F = mw,$ то за аналогією з коефіцієнтом подібності для швидкості коефіцієнт подібності для сили визначається залежністю $V_f = v_m v_l/v_f^2$. Таким же чином можуть бути знайдені коефіцієнти подібності для Інших фізичних величин.

При фізичному моделюванні експериментальні результати і висновки узагальнюються за допомогою критеріїв подібності.

Кількість таких критеріїв може бути значно меншою, ніж кількість параметрів, що описують той чи інший процес. Скоротити кількість

параметрів, що описують будь-яке явище або процес, можна шляхом їх групування у безрозмірні комплекси, які складаються з розмірних величин, виходячи з природи й умов досліджуваного явища або процесу. Ці безрозмірні комплекси називають *критеріями подібності.*

Рівність усіх однакових критеріїв подібності для двох фізичних явищ (процесів, систем) — необхідна і достатня умова їх подібності. Це визначається з пропорційності для подібних явиш, що їх описують і належать до критерію подібності. Розмірні фізичні параметри, які входять в критерії подібності, можуть значно відрізнятися між собою. Однаковими повинні бути лише безрозмірні критерії подібності, що характеризують натурний об'єкт і модель. Ця властивість подібних явищ складає основу фізичного моделювання реальних об'єктів. Якщо рівняння, що описують фізичне явище, відомі, то критерії подібності формуються шляхом приведення цього рівняння до безрозмірного критеріального виду.

У механіці для моделювання процесів використовують ряд класичних критеріїв подібності. Відомий закон Ньютона, який описує рух матеріальної точки при дії сили *F,* у диференціальній формі має вигляд

$$\frac{d^2x}{dt^2} = F/m, \qquad (3.1)$$

де m - маса матеріальної точки; x - її координата; t - час. Ураховуючи те, що символи диференціювання й інтегрування, які входять *у* початкові рівняння, можуть бути відкинуті, бо вони не мають розмірності, рівняння (3.1) можна записати в такому виді

$$\frac{x}{t^2} = \frac{F}{m}. \qquad (3.2)$$

Тепер, розділивши праву частину рівняння (3.2) на ліву, отримаємо *критерій подібності Ньютона*

$$K_N = \frac{Ft^2}{mx} = Idem,$$

Де *Idem* — відповідно однаково для всіх об'єктів, що розглядаються.

Для обертального руху

$$\frac{d^2\varphi}{dt^2} = M/J,$$

де *tp* - кут повороту; *M, J* - відповідно крутний момент і момент інерції тіла відносно осі обертання. Аналогічно критерій подібності Ньютона має вигляд

$$K_N = \frac{M\,t^2}{J\,\varphi} = Idem. \qquad (3.4)$$

Для системи матеріальних точок, між якими діє зв'язок, можна записати: якщо швидкості тіл із різними масами, що переміщуються на однакові відстані, однакові, то діючі на них сили пропорційні відповідним масам тіл. При вільному падінні тіл закон Ньютона можна записати у вигляді співвідношення

$$m\frac{d^2x}{dt} = mg,$$

де *g* - прискорення вільного падіння. Тоді для вільного падіння тіл критерій подібності має такий вираз

$$K_g = \frac{x}{gt^2} = Idem. \qquad (3.5)$$

Помноживши цей критерій на квадрат критерію гомохронності *Kho= vt/x* (критерій подібності механічного руху), отримаємо критерій подібності, який у літературі відомий під назвою *критерію Фруда*

$$K_{Fr} = \frac{\mathrm{v}^2}{\mathrm{gx}} = Idem. \qquad (3.6)$$

Визначимо критерій подібності пружних тіл. Пружну силу деформованого елемента механічної системи (наприклад, при розтягуванні пружного стержня) можна записати у вигляді

$$F = E\,S, \qquad (3.7)$$

де *E* - модуль пружності першого роду; 5 - відповідна площа поперечного перерізу деформованого елемента. Крім того, згідно із законом Ньютона

$$F = m\frac{d^2x}{dt^2}. \qquad (3.8)$$

Прирівнявши вирази сил із залежностей (3.7) і (3.8), одержимо

$$ES = m\frac{d^2x}{dt^2}.$$

Замінивши в цьому рівнянні $m = Sxp$ (де p - густина матеріалу елемента), будемо мати

$$ES = Sx\rho\frac{d^2x}{dt^2}.$$

Застосувавши правило визначення критерію подібності, отримаємо

$$K_E = \frac{Et^2}{x^2\rho} = Idem. \qquad (3.9)$$

Розділивши квадрат критерію гомохронності $K_{но}$ на критерій (3.9), будемо мати

$$\frac{K_{HO}^2}{K_E} = \frac{\mathrm{v}^2}{E/\rho}.$$

Добувши квадратний корінь із цього виразу, знайдемо критерій подібності, який має назву критерію Коші

$$K_{CO} = \frac{\mathrm{v}}{\sqrt{E/\rho}} = Idem. \qquad (3.10)$$

Величина $\sqrt{E/\rho}$ являє собою швидкість розповсюдження звукових (коливальних) хвиль у пружному середовищі.

Кожний із критеріїв подібності має свій фізичний зміст. Так, останній критерій подібності показує співвідношення між швидкістю руху тіла й швидкістю розповсюдження звукових хвиль у пружному середовищі.

Критерії подібності фізичних явищ, процесів, технічних систем та ін. незалежні один від одного і їх поєднання дає нові критерії, які відображають ті чи інші фізичні властивості.

3.2.3. Теореми подібності

Співвідношення між параметрами подібних явищ базуються на трьох теоремах подібності, в яких сформульовані необхідні і достатні умови подібності.

Перша теорема подібності встановлює необхідні умови подібності й формулюється таким чином: якщо фізичні явища подібні, то критерії подібності цих явищ рівні між собою

$$K_1 = K_2 = \ldots = K_n = Idem. \qquad (3.11)$$

Індекси 1,2,..., *n* показують номер явища.

Друга теорема подібності встановлює математичну структуру рівнянь, Ідо описують подібні фізичні явища: функціональна залежність між параметрами, що характеризують явище, може бути виражена у вигляді залежності між критеріями подібності, які сформовані з цих параметрів.

З цієї теореми випливає, що експериментальні результати необхідно обробляти у вигляді узагальнюючих безрозмірних змінних величин, а рівняння, які використовують ці результати, представляти в критеріальній формі. В цьому випадку розв'язуваная рівнянь дозволяє на основі даних єдиного експерименту проводити узагальнення при інших умовах, навіть при натурних експериментах.

Третя теорема подібності вказує на достатні умови подібності: два фізичні явища подібні, якщо вони описуються однією і тією ж системою рівнянь та мають подібні граничні умови однозначності, а їх критерії подібності чисельно рівні.

Рівняння, про які йде мова, являють собою в основному диференціальні рівняння математичної фізики. Вони представляють математичний запис фундаментальних фізичних явищ.

Багато фізичних явищ описуються тотожними диференціальними рівняннями. На цьому принципі тотожності рівнянь побудоване аналогове моделювання. Тут система диференціальних рівнянь становить математичну

модель деякого класу подібних явищ.

При інтегруванні диференціальних рівнянь отримують безмежну множину розв'язків, які задовольняють ці рівняння. Для отримання розв'язків, що враховують конкретні особливості явища, необхідно задаватись умовами однозначності. Ці умови не залежать від механізму явища, що описується диференціальними рівняннями, і задаються, виходячи з умов конкретної задачі. Одиничні явища з одними й тими ж умовами однозначності складають групу подібних явищ.

В умови однозначності входять:

1) геометричні параметри, які відображають розміри і форму предметів;

2) фізичні та механічні характеристики матеріалів предметів (коефіцієнт теплопровідності, коефіцієнт тертя, модуль пружності й ін.};

3) початкові умови, тобто стан системи в момент часу, від якого починається вивчення явища. Задаються функції невідомих змінних у координатах *x, y, z* для моменту часу, як правило, $i = 0$;

4) граничні умови, які відображають характер взаємодії тіл із навколишнім середовищем. Вони задаються деякими функціями від часу і змінних характеристик, які змінюються на поверхні тіл, наприклад, поверхневих сил, температури тощо.

Принцип подібності систем за відомими диференціальними рівняннями їх стану досить широко застосовується при моделюванні технічних систем будь-якої фізичної природи. Особливе значення цей принцип подібності має для складних механічних систем, коли диференціальні рівняння їх стану відомі, але їх не вдається проінтегрувати. Тоді будують фізичні моделі тих або інших складних механічних систем і за їх допомогою проводять експериментальні дослідження.

Розглянемо ще один приклад фізичного моделювання механічної системи з використанням критеріїв подібності, які відображають стан системи.

3.2.4. Метод аналізу розмірностей теорії подібності

Цей метод є основою теорії подібності, за допомогою якого будується фізичне моделювання.

В основі теорії розмірностей лежить принцип розмірної однорідності фізичних рівнянь: усі члени рівнянь, що описують фізичні явища, повинні мати однакову розмірність. На це вперше звернув увагу французький математик Ж. Фур'є.

Будь-який фізичний процес може бути описаний за допомогою функціональної залежності між розмірними і безрозмірними величинами. Величина, чисельне значення якої залежить від прийнятої системи одиниць, називається розмірною (наприклад: довжина, час, сила, енергія і т.д.). Якщо чисельне значення величини не залежить від системи одиниць, то вона називається безрозмірною величиною.

Сукупність фізичних величин, що зв'язані між собою або іншими залежностями, утворюють систему одиниць. Усі величини, що входять у систему одиниць, ділять на основні і похідні. За основні вибирають величини, які не залежать від інших величин даної системи. Похідні величини утворюються з основних або інших величин відповідно до фізичних законів.

У міжнародній системі одиниць SI *(CI)* прийнято сім основних одиниць: метр (м), кілограм (кг), секунда (с), ампер (А), кельвін (К), моль, кандела (кд). Допоміжними одиницями є радіан та стерадіан.

Розмірність величини визначається добутком степенів множників, які становлять основні одиниці. Формула розмірності, яку можна розглядати як характеристику фізичної природи похідної величини, в будь-якій системі одиниць може бути представлена функцією виду

$$[a]=A_1^{\alpha}\cdot A_2^{\beta}\cdot A_3^{\gamma}\cdots, \qquad (3.24)$$

де *[a]* - похідна одиниця; $A_1, A_2, A_3,...$- основні одиниці; *а, в,* у-будь-які дійсні числа.

При записі формул розмірності використовують символи довжини - *L,*

маси - M , часу - T і т. д. Наприклад, розмірності сили F і прискорення w у символьному записі будуть виглядати наступним чином

$$[F]=M\,L\,T^{-2};\;\; [w]=L\,T^{-2}.$$

Особливістю розмірних величин є те, що вони при метричних перетвореннях (переході від однієї системи одиниць до іншої) змінюють свої числові значення. На відміну від розмірних одиниць, безрозмірні одиниці не змінюють свого числового значення при такому переході.

Розглянемо структуру функціональних зв'язків між фізичними величинами, що виражають фізичний закон, незалежний від вибору системи одиниць. Нехай маємо розмірну величину я, яка є функцією розмірних величин $a_1, a_2,...,a_n$.

$$a=f(a_1,\ldots,a_k,\ldots,a_n).$$

Серед цих аргументів можуть бути виділені k величин, які будуть мати незалежні розмірності (число основних одиниць повинно бути більше або дорівнювати k). Незалежність розмірностей означає, що формула, яка виражає розмірність будь-якої з них, не може являти собою комбінацію, складену з формул розмірності для інших величин. Наприклад, розмірності довжини - L, швидкості - LT^{-1} і

енергії -$ML^2\,T^{-2}$ незалежні, а розмірності довжини - L, швидкості - $L\,T^{-1}$ і прискорення - $L\,T^{-2}$ залежні, бо

$$LT^{-2}=\left(LT^{-1}\right)^2\cdot L^{-1}.$$

Серед механічних величин є не більше від трьох з незалежними розмірностями- Приймемо k незалежних величин $a_1,...,a_к$ за основні і введемо для їх розмірностей позначення: $[a_1]=A_i$, $[a_2]=A_2,...$, $[a_k]=A_k$. Тоді відповідно до формули розмірностей (3.24) інші величини мають вигляд

$$\begin{aligned}
&[a_1]=A_1^{\alpha}\cdot A_2^{\beta}\cdot A_3^{\gamma}\cdot\ldots\cdot A_k^{\chi};\\
&\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\\
&[a_n]=A_1^{\alpha_n}\cdot A_2^{\beta_n}\cdot A_3^{\gamma_n}\cdot\ldots\cdot A_k^{\chi_n}.
\end{aligned}\qquad (3.25)$$

Перейдемо до нової системи основних одиниць b, які пропорційні

одиницям *a*. При переході від однієї системи основних одиниць (a_1, a_2, ..., a_k) до Іншої (b_1, b_2,...,b_k) змінюється не природа процесу, а числові значення розмірних величин. Структура Функціонального зв'язку f, за допомогою якого виражається фізичний закон у новій системі одиниць, має вигляд

$$b = f(b_1,\ldots,b_k,\ldots,b_n) = \\ = f(l_1 a_1,\ldots,l_k a_k,\ldots,a_n l_1^{\alpha_n} l_2^{\beta_n} l_3^{\gamma_n}\ldots). \qquad (3.26)$$

Оскільки вибір коефіцієнтів l_1, l_2,..., l_k нічим не обмежується, задамо їх у вигляді

$$l_1 = 1/a_1,\ l_2 = 1/a_2,\ \ldots,\ l_k = 1/a_k .$$

Тоді числові значення перших *k* аргументів у рівнянні (3.26) стануть рівними одиниці, тобто перетворяться в безрозмірні величини

$$b_1 = l_1 a_1 = 1,\quad b_2 = l_2 a_2 = 1,\ldots,b_k = l_k a_k = 1.$$

Також у безрозмірні величини перетворюються розмірні величини b, $b_{k+1},\ldots,b_n$. їх числові значення визначаються залежностями:

$$b = \frac{a}{a_1^{\alpha} a_2^{\beta} a_3^{\gamma}\ldots a_k^{\chi}} = K; \\ b_{k+1} = \frac{a_{k+1}}{a_1^{\alpha_{k+1}} a_2^{\beta_{k+1}} a_3^{\gamma_{k+1}}\ldots a_k^{\chi_{k+1}}} = K_1; \qquad (3.27) \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ b_n = \frac{a_{k+1}}{a_1^{\alpha_n} a_2^{\beta_n} a_3^{\gamma_n}\ldots a_k^{\chi_n}} = K_{n-k} .$$

Таким чином, у результаті метричних перетворень отримана безрозмірна система одиниць фізичних величин, у яких залежність (3.26) з урахуванням виразів (3.27) представляється у вигляді

$$K = f(1,\ldots,1,K_1,\ldots,K_{n-k})$$

або

$$K = \varphi(K_1,\ldots,K_{n-k}). \qquad (3.28)$$

З останнього рівняння можна зробити декілька висновків. По-перше, всяке фізичне співвідношення між розмірними величинами можна сформулювати як співвідношення між безрозмірними величинами. По-друге, зв'язок між n+1 розмірними величинами a, $a_1,\ldots,a_n$, незалежний від вибору

системи одиниць, приймає вигляд співвідношення між $n+l-k$ безрозмірними величинами $K,K_1,..,K_{n-k}$. При цьому число аргументів зменшилось на k, що відповідає кількості аргументів, які мають незалежні розмірності.

Якщо розмірна величина є також функцією безрозмірних аргументів $a_{n+1},...,a_s$, то рівняння фізичного процесу приймає вигляд

$$a = \varphi(a_1,\ldots,a_n,a_{n+1},\ldots,a_s). \qquad (3.29)$$

Для безрозмірних аргументів можна записати, що

$$[a_{n+1}] = [a_{n+2}] = \ldots = [a_s] = 1.$$

При переході до нової системи одиниць безрозмірні величини зали шаються без змін:

$$\begin{aligned} b_{n+1} &= a_{n+1} = K_{n+1-k}, \\ &\ldots\ldots\ldots\ldots\ldots \\ b_s &= a_s = K_{s-k}. \end{aligned} \qquad (3.30)$$

У результаті метричних перетворень вираз (3.29) приймає вигляд

$$K = \psi(K_1,\ldots,K_{n-k},K_{n+1-k},\ldots,K_{s-k}). \quad (3.31)$$

В отриманому рівнянні зв'язок між $s+1$ розмірними і безрозмірними величинами приймає вигляд зв'язку між $s + 1 — k$ безрозмірними комплексами.

При використанні методу аналізу розмірностей важливим елементом є техніка приведення фізичних рівнянь до безрозмірного виду. Нехай розмірна фізична величина a є функцією s аргументів a_i,

$$a = f(a_1,\ldots,a_k,\ldots,a_n,\ldots,a_s), \qquad (3.32)$$

причому n з них розмірні, а $s—n$ безрозмірні. Спочатку виписують розмірності всіх величин. З числа a аргументів виділяють k аргументів із незалежними розмірностями. За допомогою формул (3.27) утворюють безрозмірні комплекси $K,K_1,...,K_{n-к}$, які доповнюють безрозмірними аргументами (3.30) в кількості $s- n$. Вважаючи, що $K_{n-k+1} = a_{n+1},... ...,K_{s-к} = a$, початкове рівняння (3.32) приводять до безрозмірного виду

$$K = \psi(K_1,\ldots,K_{n-k},K_{n-k+1},\ldots,K_{s-k}). \quad (3.33)$$

Розглянемо приклад застосування методу аналізу розмірностей Для

дослідження механічних систем.

Метод аналізу розмірностей показує вплив як кожного з аргументів окремо, так І їх сумісний вплив на кінцевий результат. Разом з тим аналізу розмірностей для отримання кінцевих результатів недостатньо. Конкретний вигляд функції процесу може бути отриманий дослідним шляхом або теоретично. При користуванні цим методом головним є виявлення основних факторів, які визначають суть явища або процесу, до вивчаються.

За допомогою методу аналізу розмірностей відкрити фізичні закони неможливо. Цей метод розглядається як засіб для упорядкування наших уявлень про характер діючих в природі закономірностей.

3.3. Математичне моделювання

3.3.1. Основні поняття математичного моделювання

Побудова математичних моделей являє собою основу теорії систем. Це центральний етап у дослідженні, проектуванні або керуванні будь-якої системи. Від якості математичної моделі залежить доля всіх розглянутих етапів.

У загальному випадку під математичною моделлю технічної або іншої системи розуміють будь-яке співвідношення, яке відображає з необхідною точністю поведінку реальної системи в реальних умовах. Математична модель концентрує записану мовою математичних співвідношень сукупність наших знань, уявлень і гіпотез про відповідний об'єкт, явище, процес або систему. Оскільки ці знання ніколи не бувають абсолютними, а гіпотези можуть інколи цілеспрямовано не враховувати деякі фактори, то модель лише наближено враховує поведінку реальної системи.

Математичну модель можна розглядати як деякий оператор, який ставить у відповідність системі внутрішніх параметрів технічної системи a_1, a_2, ..., a_m сукупність функціонально зв'язаних між собою зовнішніх

параметрів y_1, y_2, ..., y_n. Вигляд функціонального зв'язку залежить від фізичного принципу дії системи, а зміст понять внутрішніх та зовнішніх параметрів системи визначається її фізичною суттю і способом використання.

Таким чином, для складних технічних систем неможливо отримати абсолютно подібні (ізоморфні) математичні моделі. Шляхом формалізації системи одержують спрощену модель, яка відображає основні властивості й не враховує другорядні.

Стан технічної системи в будь-який довільний момент часу I із за, даного інтервалу $[t_0, t_1]$ можна охарактеризувати набором величин x_1, X_2,..., x_l — характеристиками стану системи. При функціонуванні технічної системи ці характеристики приймають значення, що є функціями часу, тобто $\{x_1(t),...,X_l(t)\}=X(t)$, де-X(t) — вектор стану системи Проекції цього вектора можна розглядати як координати точки в n-мірному фазовому просторі, а процес функціонування системи — як деяку фазову траєкторію.

На систему може діяти вектор вхідних дій $U(t) = \{u_1(t),...,u_m(t)\}$, від яких залежать характеристики стану. Система характеризується також набором власних параметрів $A=\{a_1,...,a_p\}$, що в стаціонарній системі е константами, а в нестаціонарній — функціями часу. На систему можуть діяти деякі випадкові фактори | = {£[,...,£$_v$}. Вона може мати також ряд виходів $Y=\{Y_1(t),...,Y_n(t)\}$.

Таким чином, математична модель технічної системи — це сукупність співвідношень (формул, нерівностей, рівнянь, логічних співвідношень і т. д.), які визначають характеристики стану системи залежно від її параметрів, вхідних дій, випадкових факторів, початкових умов та часу. Початкові умови — це значення характеристик системи в початковий момент часу t_0 : X_{10} , X_{20} ,..., X_{n0} .

Так, для динамічних технічних систем математичними моделями можуть бути диференціальні рівняння виду

$$\dot{x} = f(t, x, u, \xi). \qquad (3.37)$$

За допомогою таких моделей, коли задано початковий стан, визначають

траєкторію процесу.

Залежно від специфіки зв'язків характеристик стану системи з її параметрами і вхідними сигналами розрізняють;

1) детерміновані моделі, в яких у заданий момент часу характеристики стану однозначно визначаються через вказані величини. В цих моделях £ = 0;

2) імовірнісні (стохастичні) моделі, в яких за допомогою математичних співвідношень можна визначити лише розподілення характеристик стану системи за заданими імовірнісними характеристиками (розподіленнями) її параметрів, вхідних сигналів, початкових умов.

За ознакою подальшого використання математичні моделі розділяють на аналітичні та імітаційні. Аналітичні моделі забезпечують достатньо високий ступінь деталізації опису системи, однак не завжди дають змогу отримати висновки загального характеру про її функціонування. У випадку застосування імітаційних моделей можуть ураховуватися такі особливості складних технічних систем, як наявність в одній і тій же системі елементів неперервної й дискретної дії, нелінійні співвідношення будь-якого характеру, що описують зв'язки між елементами, вплив багаточисельних випадкових факторів складної фізичної природи тощо.

3.3.2. Побудова математичних моделей

У практичних задачах, на відміну від задач чисто математичних, не завжди буває з самого початку ясно, що задано І що необхідно довести або визначити. До таких задач належить і задача побудови математичних моделей технічних систем. Звичайно задається реальна технічна система. Необхідно побудувати її математичну модель для тієї чи іншої мети.

Розв'язування таких практичних задач починається зі збору фактів та даних наукових спостережень. На їх основі проводиться формалізація технічної системи і будується її математична модель, тобто виділяються її найбільш суттєві риси та властивості й проводиться їх опис за допомогою

рівнянь і формул.

Розглянемо основні етапи побудови математичних моделей реальних технічних систем (рис. 3.5).

Етап 1. При виясненні і постановці задачі на фізичному рівні проходить процес схематизації та ідеалізації технічної системи, тобто виділення її суттєвих факторів, що впливають на функціонування системи. Деякі риси і фактори системи можуть виявитися важливими, інші — несуттєвими.

Етап 2. Після виявлення суттєвих факторів ставляться задачі моделювання й вибирається схема взаємодії між елементами системи. Для динамічних систем будується динамічна модель, яка відображає суттєві фактори. Здійснюється переведення необхідних характеристик на мову математичних понять І величин. Складається система параметрів, які описують основні фактори, й здійснюється формування співвідношень та рівнянь між цими параметрами і величинами. Це найбільш складна й важка стадія процесу моделювання. Тут використовують фундаментальні фізичні закони і принципи.

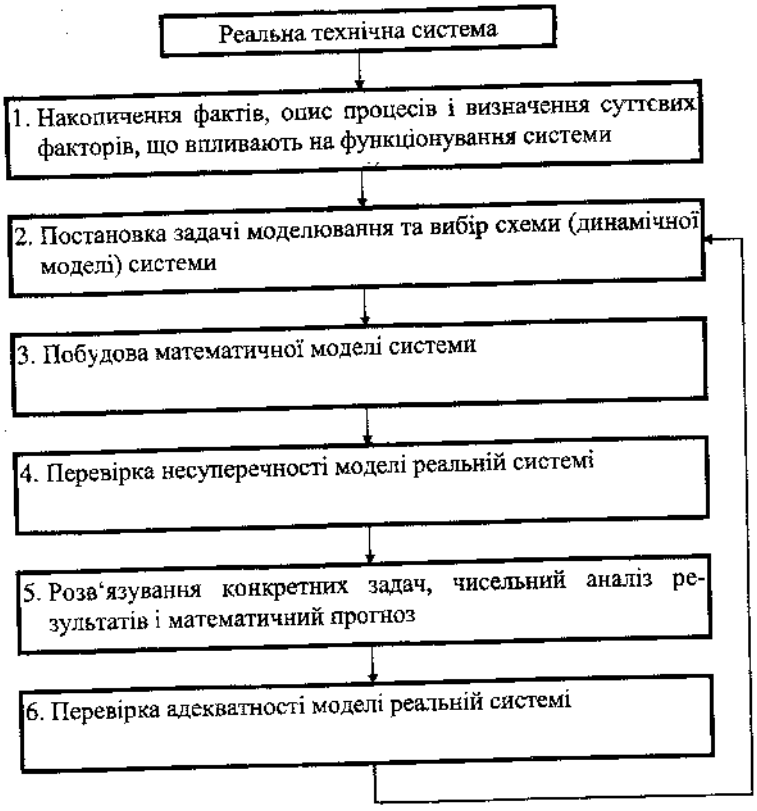

Рис. 3.5. Етапи побудови математичної моделі

Етапи 3. 4. Після побудови моделі (етап 3) необхідно проводити перевірку суперечності моделі реальній системі і конкретності постановки задачі. Тут можна використати досить просте й завжди ефективне правило фізичної розмірності всіх членів рівняння, які складені за законами збереження.

Етапи 5. 6. Перевіряється справедливість моделі за результатами розв'язування теоретичної задачі у відповідності з математичною моделлю, які зіставляються з реальними результатами технічної системи. На основі цих результатів перевіряється адекватність математичної моделі реальній системі. Глибина відображення моделлю реальної технічної системи залежить від мети дослідження.

Відповідно до принципів ієрархії математичних моделей кожна модель нижчого рівня не повинна суперечити моделі вищого рівня. На самому нижчому рівні будують математичні моделі конкретних процесів і найпростіших явищ технічної системи.

Математична модель — це результат формалізації реальної технічної системи. Процес формалізації системи при побудові її моделі складається з трьох основних етапів:

1) складання змістового опису реальної системи, тобто побудова дескриптивної моделі;

2) побудова формалізованої схеми;

3) розроблення математичної моделі.

Дескриптивна модель становить першу спробу словесного опису закономірностей, що характеризують функціонування технічної системи, а також змістової постановки задачі або формулювання мети дослідження. Для побудови такої моделі необхідне вивчення системи шляхом спостереження за нею і фіксації деяких кількісних характеристик.

При побудові моделі системи, що проектується, словесний опис складається на основі досвіду, а також на основі спостережень за аналогічними, реально існуючими системами.

Дескриптивна модель, як правило, складається спеціалістами в конкретній галузі техніки без активної участі математиків за результатами дослідження технічної системи. Однак вона повинна обов'язково містити перелік залежностей, які підлягають оцінці, а також перелік факторів, котрі повинні бути враховані при побудові моделі. В дескриптивну модель включаються початкові дані у вигляді таблиць, графіків, початкових умов. Змістовий опис самостійного значення не має, але служить основою для подальшої формалізації технічної системи.

Формалізована схема — це проміжний етап між словесним описом і математичною моделлю. Вона реалізується в тому випадку, коли неможливий з деяких причин безпосередній перехід від дескриптивної моделі до математичної. При побудові формалізованої схеми необхідно вибирати сукупність характеристик стану й параметрів технічної системи. За характеристики стану бажано вибирати такі функції, які забезпечують зручну можливість визначення невидних характеристик і дозволяють отримати достатньо просту Математичну модель.

Вибір параметрів, що характеризують технічну систему, визначається тими факторами, які необхідно враховувати при побудові математичної моделі. На етапі побудови формалізованої схеми технічної системи повинна бути чітко сформульована математична мета дослідження.

Подальше перетворення формалізованої схеми в математичну модель здійснюється практично без притоку допоміжної Інформації.

Для динамічних технічних систем, тобто систем, які змінюють свій стан з часом, формалізованою схемою виступає динамічна модель. Найбільш широке застосування динамічні моделі отримали при побудові математичних моделей механічних систем. Спочатку будується динамічна модель технічної системи, а потім на її базі формальними методами складається математична модель, тобто наявність динамічної моделі однозначно визначає математичну модель технічної системи.

3.3.3. Динамічна модель механічної системи

При переході від реальної механічної системи (машини) до її динамічної моделі нехтують тими фізичними факторами, які несуттєві для даного розрахунку або дослідження. В загальному випадку при складанні динамічної моделі механічної системи необхідно враховувати зосереджені маси, розподілені маси по довжині елементів, пружність елементів, залежності рушійних та гальмівних сил двигунів від частоти обертання ротора, зміну приведених мас І т. д. У кожному конкретному випадку одні фізичні фактори є головними, а інші — другорядними.

Динамічна модель повинна задовольняти дві головні вимоги:

1) бути в необхідній мірі адекватною реальній механічній системі й, наскільки це можливо, відображати основні її фізичні властивості;

2) бути не дуже складною, щоб розв'язування не було досить трудомістким.

3.3.4. Методи побудови математичних моделей механічних систем

На основі отриманої динамічної моделі формальними методами може бути побудована математична модель будь-якої механічної системи. Математичні моделі механічних систем становлять, як правило, диференціальні рівняння руху або взаємодії окремих елементів.

Для отримання диференціальних рівнянь руху механічних систем при відомих їх динамічних моделях використовуються три основних методи: 1) метод рівноваги з використанням принципу Даламбера; 2) принцип можливих переміщень; 3) принцип Гамільтона-Остроградського.

Розглянемо більш детально кожний із цих методів.

Метод рівноваги. Рівняння руху будь-якої механічної системи при наявності її динамічної моделі — це вираз другого закону Ньютона, який встановлює, що швидкість зміни імпульсу будь-якої маси дорівнює діючій на

неї силі. В математичній формі це записується у вигляді наступного диференціального рівняння:

$$\bar{F}(t) = \frac{d}{dt}\left(m\frac{d\bar{r}}{dt} \right), \qquad (3.50)$$

де *F(t)*- вектор прикладеної сили; r - радіус-вектор координат центра мас маси *m; t* - координата часу.

Для більшості задач динаміки машин і механізмів масу можна розглядати незмінною в часі. Тоді рівняння (3.50) приймає вигляд

$$\bar{F}(t) = m\frac{d^2\bar{r}}{dt^2} = m\ddot{\bar{r}}(t).$$

Отримане рівняння виражає умову рівності сили добутку маси на прискорення

$$\bar{F}(t) - m\ddot{\bar{r}}(t) = 0. \qquad (3.51)$$

У рівнянні (3.51) другий доданок називають силою інерції, яка здійснює опір прискоренню маси.

Принцип Даламбера *(маса викликає силу інерції, пропорційну її прискоренню і протилежно йому спрямовану)* широко застосовується в задачах динаміки машин, оскільки дає змогу вивести рівняння руху на основі умов динамічної рівноваги. Сила F*(t)* може включати в себе різні види сил, що прикладені до маси: силу пружного опору, яка направлена в протилежному переміщенню напрямку; силу затухання, яка здійснює опір швидкості переміщення, і незалежні зовнішні сили. Якщо ввести силу інерції, що здійснює опір прискоренню маси, то рівняння руху виражають умову рівноваги всіх сил, які прикладені до маси. Принцип Даламбера розглядає рівновагу окремо взятої маси з прикладенням до неї всіх діючих сил, сили інерції та реакцій зв'язку з іншими масами. Для більшості простих динамічних моделей механічних систем указаний метод виводу рівнянь руху найбільш зручний.

Принцип можливих переміщень. Коли конструктивна схема механічної системи достатньо складна і містить ряд взаємодіючих тіл кінцевих розмірів,

безпосереднє виведення умов рівноваги всіх діючих на систему сил ускладнюється, Змінні сили часто виражаються через переміщення по узагальнюючих координатах, але записати умови їх рівноваги досить складно. В цьому випадку для виведення рівнянь руху замість умов рівноваги використовують принцип можливих (віртуальних) переміщень.

Цей принцип формулюється наступним чином: *Якщо система,котра знаходиться в рівновазі під дією декількох сил, отримує можливе переміщення, тобто будь-яке переміщення, яке задовольняє крайовим умовам, то повна робота всіх сил на цьому переміщенні дорівнює нулю.*

Згідно з цим принципом рівність нулю роботи сил на можливому переміщенні системи еквівалентна умові рівноваги. Суттєва перевага цього принципу полягає в тому, що складові робіт сил на можливих переміщеннях — скалярні величини і можуть додаватись алгебраїчне, а сили, які діють на елементи динамічної моделі, становлять вектори і можуть додаватися тільки за правилами векторного аналізу.

Принцип Гамільтона-Остроградського. Цей метод не вимагає векторних рівнянь рівноваги, бо він використовує скалярні величини енергії у варіаційній постановці. *Суть цього методу полягає в тому, що для неконсервативних механічних систем справедливе варіаційне рівняння*

$$\int_{t_0}^{t_1}(\delta T+\delta A)dt=0, \qquad (3.56)$$

де t_0, t_1- початковий і кінцевий моменти часу руху системи; *ST*-варіація кінетичної енергії; b*A*- елементарна робота сил, прикладених до системи, при переході від прямого до обхідного шляху, який має з прямим шляхом спільні початкові й кінцеві умови.

Якщо система консервативна, то b*A* = - *bП* (де П - потенціальна енергія системи) і b*T* + *bA*= *b(T-П)=bL.* У випадку консервативної системи принцип Гамільтона-Остроградського полягає в тому, що

$$\delta\int_{t_0}^{t_1}L\,dt=0. \qquad (3.57)$$

Інтеграл

$$I_L = \int_{t_0}^{t_1} L\,dt$$

називається дією, за Гамільтоном-Остроградським.

Застосування цього принципу можна здійснювати і в іншій формі

$$\int_{t_0}^{t_1} \left[\delta\left(T - \Pi\right) + \delta A_1\right] dt = 0. \qquad (3.58)$$

У цьому випадку консервативні сили (гравітаційні й пружні) входять у вираз потенціальної енергії, а bA_1 становить елементарну роботу неконсервативних сил (рушійних і сил опору при переміщенні системи).

Застосування припилу Гамільтона-Остроградського у формі (3.58) дає можливість спростити врахування консервативних сил, таким чином надати принципу більший формалізм.

Принцип Гамільтона-Остроградського можна покласти в основу наближених методів розв'язування задач динаміки машин, які широко застосовуються в теорії пружності й при розв'язуванні складних задач теорії коливань.

Усі три методи отримання диференціальних рівнянь руху механічних систем рівнозначні, і ці методи для однієї й тієї ж динамічної моделі приводять до одного і того ж результату. Звичайно вибір методу для будь-якої конкретної механічної системи залежить від типу динамічної моделі та визначається самим дослідником.

Для отримання необхідних результатів диференціальні рівняння руху механічної системи підлягають інтегруванню з метою визначення характеристик стану (переміщень, швидкостей і прискорень) окремих елементів у функції часу.

3.3.5. Ідентифікація як метод побудови математичних моделей

Задачу ідентифікації можна сформулювати наступним чином за результатами спостережень за вхідними і вихідними змінними технічної системи побудувати її модель. При цьому система знаходиться в

нормальному режимі функціонування. Математично задача формулюється так: якщо технічна система описується деяким оператором A_t, апріорі невідомим, то, маючи заміряні параметри входу й виходу, необхідно побудувати модель оператора A_t, оптимальну за деяким критерієм.

Розглянемо взаємодію системи, яка ідентифікується, з середовищем (рис.3.11). Ця взаємодія проходить по каналах $\vec{Z}$ і $\vec{Y}$. По каналу $\vec{Z}$ середовище впливає на технічну систему (ТС), а по каналу у ТС діє на середовище.

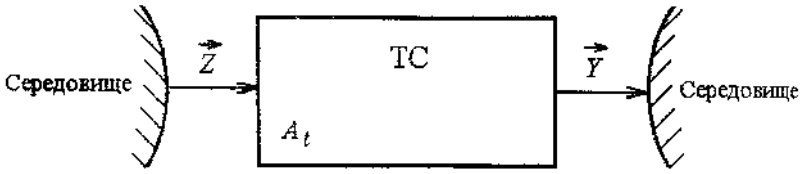

Рис. 3.11. Взаємодія технічної системи з середовищем

Задача ідентифікації зводиться до визначення оператора A_t, який зв'язує вихід ТС $\vec{Y} = A_t(\vec{Z})$.Оскільки досить часто відсутня модель середовища, що діє на ТС, то вхід можна розглядати як випадкову функцію часу $\vec{Z} = \vec{Z}(t)$, статистичні властивості якої в загальному випадку невідомі. Однак відомі спотереження входу і виходу ТС, тобто реалізації функції $\vec{Z}(t)$ та $\vec{Y}(t)$. Неспостережувані функції $\xi(t)$, які можуть діяти на ТС, розглядаються як випадкові дії, що утруднюють визначення оператора A_t^1.

Нехай Z_1 ,...,Z_n - спостереження входу ТС, а, Y_1,...,Y_n-відповідні їм спостереження її виходу в дискретні моменти часу t_1,...,t_N. Ці спостереження зв'язані невідомим опереатором ТС A_t тобто $Y_i=A_t(Z_i)$, i = 1, 2, ... , N. Задача ідентифікації полягає в попові (синтезі) модельного оператора A_t^1 тобто в отриманні гілки A_t за спостереженнями Z_i і Y_i в дискретні моменти часу t_i.

Таким чином, ідентифікація — це синтез оптимального модельного оператора A_t^1 ТС із використанням результатів спостережень за й вхідними й вихідними змінними.

Згідно з сучасною теорією можна провести наступну класифікацію ідентифікації:

1)за кінцевим результатом ідентифікації — структурна та параметрична;

2) за способом вивчення ТС ідентифікації — активна і пасивна;

3) за типом моделі, що ідентифікується, — лінійна й нелінійна, детермінована і стохастична, з неперервним та дискретним часом, стаціонарна і нестаціонарна, одномірна й багатомірна, статична й динамічна, з дискретними і розподіленими параметрами.

Успіх ідентифікації ТС суттєво залежить від співвідношення двох факторів: обсягу апріорної інформації про структуру ТС та обсягу вимірювальної інформації. Обидва види інформації необхідні яри синтезі моделі, однак вони відіграють різні ролі. Апріорна інформація допомагає визначити структуру моделі, тобто її вид (число входів і виходів, характер зв'язку між ними). Цю процедуру називають структурною ідентифікацією.

Однак структура моделі — це ще не сама модель, і для визначення її параметрів необхідно мати результати вимірювань. Задачу визначення параметрів моделі за результатами роботи ТС при заданій структурі моделі називають параметричною ідентифікацією. Наприклад, є певна ТС й відома система рівнянь, яка її описує. Необхідно визначити тільки коефіцієнти рівнянь.

Першими і найпростішими системами, які були ідентифікованими, виявилися статичні нестохастичні системи, тобто регулярні функції, що зв'язують входи та виходи ТС. Ця обставина створила перший підхід теорії ідентифікації, який з'явився в математичному аналізі у вигляді теорії наближення функцій многочленами і веде свій початок від праць П.Л.Чебишева. Цей напрямок пов'язаний з представленням функції у вигляді розвинення по деякій системі функцій (наприклад, поліномів). Теорія наближення має дві гілки — теорію апроксимації та теорію інтерполяції. Остання характерна тим, що інтерпольована функція збігається з початковою в заданому наборі точок.

Для ідентифікації стохастичних ТС застосовують методи математичної статистики, що дало початок теорії оцінювання. Основною задачею цієї теорії є оцінка параметрів стохастичної системи за спостереженнями випадкових дій. Іншим напрямом математичної статистики для цілей ідентифікації статичних стохастичних ТС стала теорія планування експериментів, яка розглядає активні експерименти з метою підвищення ефективності ідентифікації.

Третім підходом до розв'язування задач ідентифікації є методи теорії систем автоматичного керування. Ця теорія дала життя спеціальним методам ідентифікації динамічних ТС керування в режимі експлуатації при дії випадкових факторів. Саме до цих систем уперше був застосований термін "ідентифікація".

При структурній ідентифікації обсяг апріорної інформації про ТС досить обмежений. Тому необхідно розв'язати наступні задачі:

1) виділення ТС із середовища;

2) вибір класу моделей ТС;

3) визначення характеру зв'язку між входом і виходом моделі ТС;

4) оцінка ступеня та форми впливу вхідних змінних на вихідні. визначення раціональної кількості вхідних та вихідних змінних, що враховуються в моделі;

5) визначення можливості представлення моделі з необхідною точністю в класі лінійних операторів.

Розглянемо деякі способи ідентифікації на прикладі одномірної ТС (рис. 3.12) із зосередженими параметрами. Реальна ТС описується оператором A_t , тобто у формі $y(t)=A_t(Z(t))$, який неможливо знайти, але можна зробити його оцінку. Уточнюючи результат оцінки, отримують ідентифікацію. Застосовуючи деякий алгоритм ідентифікації (AI), необхідно побудувати модель $y^1(t)= A_t^1(Z(t))$ з оптимальним оператором A_t^1, достатньо близьким до A_t, який забезпечує при однаковому вхідному сигналі Z(t) близькість вихідних сигналів y(t) і $y^1(t)$

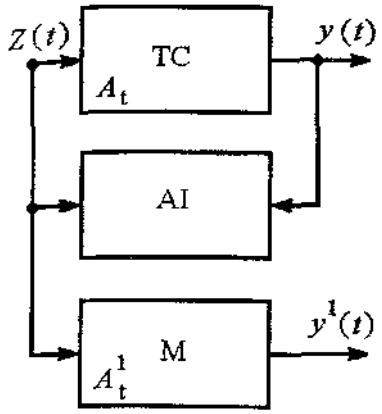

Рис. 3.12. Схема ідентифікації одномірної технічної системи

Зазначимо, що вказана близькість досить відносна, бо оператори A_t й A_t^1 можуть мати різну структуру, можуть бути сформульовані на різних мовах і мати різне число входів. Саме тому близькість операторів безпосередньо оцінити важко або неможливо, бо про оператор ТС досить часто мало що відомо. В зв'язку з цим необхідно оцінювати близькість операторів за їх реакціями на одну й ту ж вхідну дію Z(t), тобто по виходах ТС $y(t) = A_t(Z(t), \xi(t))$ та моделі $y^1(t) = A_t^1 (Z(t), \xi^1(t))$. У загальному випадку Z(t) й y(t) можуть бути як детермінованими, так і випадковими функціями часу.

Оптимальний оператор A_t^1 моделі шукається за деяким критерієм, який зв'язаний із вихідною змінною y(t), наприклад, для детермінованих функцій

$$\left[y(t) - y^1(t)\right]^2_{\max} \to \min; \qquad (3.98)$$

для випадкових функцій часу

$$M\left\{\left[y(t) - y^1(t)\right]^2\right\} \to \min. \qquad (3.99)$$

Для розв'язування подібних задач уводиться поняття функції втрат (функції нев'язки) $\rho\left[y(t), y^1(t)\right]$, яка в будь-який фіксований момент часу t залежить від виходу ТС і моделі та не залежить від операторів. Це скалярна функція двох векторних аргументів — виходів ТС і моделі. Найбільш часто функція втрат використовується У вигляді середнього за t квадрата відхилення

$$\rho\left[y(t), y^1(t)\right] = M\left\{\left[y(t) - y^1(t)\right]^2\right\}. \quad (3.100)$$

Критерієм оптимальності оператора моделі A_t^1 є мінімум функції втрат ρ.

Відомо [19], що оптимальну оцінку оператора ТС за критерієм мінімуму середнього квадрата відхилення в класі всіх можливих операторів дає умовне математичне сподівання вихідної змінної відносно вхідної (регресія вихідної змінної y(t) по вхідній z(s)):

$$y^1(t) = A_t^1\left[z(s)\right] = M\left[y(t)/z(s)\ ,\ s \in T\right]. \quad (3.101)$$

Звичайно оптимальний оператор шукають у класі лінійних операторів, для яких може бути застосований принцип суперпозиції

$$A_t\left[\sum_{i=1}^{n} a_i z_i(t)\right] = \sum_{i=1}^{n} A_t\left[a_i z_i(t)\right].$$

Методи ідентифікації умовно ділять на пасивні та активні. Типовими *пасивними* методами є методи автоматичного керування, які використовуються для ідентифікації пристроїв регулювання динамічних ТС. При дослідженні процесів (наприклад, технологічних) широко застосовують активні експериментально-статистичні методи планування експериментів [20], а для аналізу детермінованих статичних ТС використовують теорію наближення функцій (наприклад, методи апроксимації й інтерполяції).

Покажемо практичні можливості застосування активних методів ідентифікації на прикладі побудови регресивних моделей ТС. Ці моделі будуються із застосуванням методів планування експеримента. Основний недолік моделей полягає в тому, що вони є локальними, тобто адекватні ТС у порівняно вузькому діапазоні факторів. Обробка експериментальних даних здійснюється методами класичного регресивного аналізу [20].

У загальному випадку, якщо є ряд факторів $\{z_1, z_2, ..., z_k\} = \vec{Z}$, що діють на ТС, то відгук y є емпіричною функцією цих факторів, тобто $y = \varphi(z_1, z_2, ..., z_k)$, апріорі невідомою. Будемо будувати модель реакції системи (регресивну модель) у вигляді деякого полінома відносно цих факторів

$$y = \beta_0 + \sum_{j=1} \beta_j z_j + \sum_{j,i=1}^{k} \beta_{ji} z_l z_j + \ldots, \quad (3.112)$$

де β_0- вільний член,

$$\beta_j = \frac{\partial \varphi}{\partial z_j}\bigg|_{\vec{Z}=0}; \quad \beta_{ji} = \frac{\partial^2 \varphi}{\partial z_j z_i}\bigg|_{\vec{Z}=0}.$$

Тут β_j ураховує лінійний ефект, а β_{ji}, — квадратичний ефект (i = j) та ефект взаємодії між факторами (i≠j). Ці коефіцієнти неможливо точно визначити, бо невідомий аналітичний вираз для функції φ, а можливо дати лише їх оцінки $b_0 = \beta_0^1$, $b_j = \beta_j^1$,..., тобто фактично регресивну модель можна отримати у вигляді

$$y^1 = b_0 + \sum_{j=1}^{k} b_j z_j + \sum_{i,j=1}^{k} b_{ji} z_j z_i + \ldots = f(\vec{Z}, b_0, b_1, \ldots).$$

Перед побудовою регресивної моделі необхідно обмежитись якимось ступенем полінома, а також корисно побудувати експериментальну лінію регресії, яка дозволяє зробити висновок про форму моделі і дає допоміжну можливість визначити необхідну ступінь полінома.

Виробивши вид моделі, здійснюють оцінку коефіцієнтів регресії β_0, β_1, ... , застосовуючи метод найменших квадратів. Шукають мінімум функціонала

$$\Phi = \sum_{i=1}^{N} \left(y_i - y_i^1\right)^2, \quad (3.113)$$

де N - об'єм вибірки; y_i виміряне значення виходу; $y_i^1 = f(\vec{Z}_i, b_0, b_1, \ldots)$ - значення виходу, що передбачено моделлю.

Якщо позначити через l число коефіцієнтів β у рівнянні (3.112), то s= = N – l являє собою число ступенів свободи Функціонал (3.113) буде мати мінімум, якщо $\partial \Phi / \partial b_0 = 0$, $\partial \Phi / \partial b_1 = 0$,..., або

$$2\sum_{i=1}^{N}\left[y_i - f\left(\vec{Z}_i, b_0, b_{1,\ldots}\right)\right]\frac{\partial f\left(\vec{Z}_i, b_0, b_{1,\ldots}\right)}{\partial b_0} = 0;$$

$$2\sum_{i=1}^{N}\left[y_i - f\left(\vec{Z}_i, b_0, b_{1,\ldots}\right)\right]\frac{\partial f\left(\vec{Z}_i, b_0, b_{1,\ldots}\right)}{\partial b_1} = 0;$$

.....

$$2\sum_{i=1}^{N}\left[y_i - f\left(\vec{Z}_i, b_0, b_{1,\ldots}\right)\right]\frac{\partial f\left(\vec{Z}_i, b_0, b_{1,\ldots}\right)}{\partial b_l} = 0.$$

Перетворивши цю систему до нормальної форми, отримаємо

$$\begin{aligned}
&\sum_{i=1}^{N} y_i \frac{\partial f\left(\vec{Z}_i, b_0, b_{1,\ldots}\right)}{\partial b_0} - \sum_{i=1}^{N} f\left(\vec{Z}_i, b_0, b_{1,\ldots}\right)\frac{\partial f\left(\vec{Z}_i, b_0, b_{1,\ldots}\right)}{\partial b_0} = 0;\\
&\sum_{i=1}^{N} y_i \frac{\partial f\left(\vec{Z}_i, b_0, b_{1,\ldots}\right)}{\partial b_1} - \sum_{i=1}^{N} f\left(\vec{Z}_i, b_0, b_{1,\ldots}\right)\frac{\partial f\left(\vec{Z}_i, b_0, b_{1,\ldots}\right)}{\partial b_1} = 0; \quad (3.114)\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
&\sum_{i=1}^{N} y_i \frac{\partial f\left(\vec{Z}_i, b_0, b_{1,\ldots}\right)}{\partial b_l} - \sum_{i=1}^{N} f\left(\vec{Z}_i, b_0, b_{1,\ldots}\right)\frac{\partial f\left(\vec{Z}_i, b_0, b_{1,\ldots}\right)}{\partial b_l} = 0.
\end{aligned}$$

Система алгебраїчних рівнянь (3.114) містить стільки рівнянь, скільки невідомих коефіцієнтів b_0 ,b_1 ,... . Розв'язавши цю систему при відомій структурі функції f, можна визначити коефіцієнти b_0 , b_1 , ..., b_l .

Оскільки функціонал Ф>0, то він обов'язково буде мати мінімум.

3.4. Адекватність моделі і технічної системи

3.4.1. Методи спрощення моделей

Як правило, процеси функціонування реальних технічних систем (ТС) є настільки складними, що виникає потреба спрощення їх моделей. Найбільш поширеними є наступні методи спрощення моделей:

1) розчленування складних ТС на ряд більш простих підсистем (декомпозиція) [21];

2) виділення суттєвих властивостей та дій і врахування інших (несуттєвих) факторів у параметричній формі (метод макромоделювання);

3} лінеаризація нелінійних процесів у деякій області зміни змінних загальноприйнятим методом малих відхилень;

4) приведення систем із розподіленими параметрами до систем із зосередженими параметрами;

5) нехтування динамічними властивостями процесів. Розглянемо деякі з перерахованих методів спрощення моделей більш детально.

У загальному випадку кінцевою метою декомпозиції є розчленування простору змінних $\{y_1, y_2, .., y_q, z_1, z_2, .., z_p, v_1, v_2, .., v_r, f_1, f_2, .., f_s\}$, де $\vec{V} = \{v_1, v_2, ..., v_r\}$, $\vec{F} = \{f_1, f_2, ..., f_s\}$ - відповідно спостережувані й неспостережувані (тобто неконтрольовані) дії на систему. Якщо в такій системі будь-який вихід має зв'язок з іншими виходами, то декомпозиція практично неможлива, а якщо такого зв'язку немає, то модель може бути розчленована на таку кількість моделей, скільки, існує блоків-виходів, між якими немає зв'язку. Розглянемо приклад розчленування загальної моделі на систему більш простих еквівалентних моделей.

При використанні методу макромоделювання в початковому Просторі змінних залишаються (тобто враховуються) тільки ті з них, які значно впливають на вихідні змінні. Інші невраховані змінні можуть бути враховані в параметричній формі шляхом зміни коефіцієнтів прц врахованих змінних або шляхом уведення вільних членів.

При побудові спрощених моделей із урахуванням тільки суттєвих впливів широко використовується метод адаптивної моделі, тобто моделі, коефіцієнти якої підставляються таким чином, шоб деяка міра розходження (нев'язки) виходів моделі і реальної ТС приймала допустимі (мінімальні) значення. Для цього використовують критерії мінімізації нев'язок. При цьому ті змінні, які стабілізуються й не приводять до зміни вихідних змінних, у моделі не відображаються. Структура спрощеної моделі називається макромоделлю. яка для k-ої вихідної змінної має вигляд

$$\varphi_k\left(y_1, \ldots, y_q, z_1, \ldots, z_p, v_1, \ldots, v_r, f_1, \ldots, f_s\right) = 0. \quad (3.119)$$

Маючи початкову повну модель (3.119), можна оцінити ступінь впливу на вихідну змінну y_k тієї або Іншої дії шляхом визначення похідних від y_k, тобто $\partial y_к/\partial z_j$, $\partial y_k/\partial \upsilon_i$, $\partial y_k/\partial f_k$. Для цього необхідно тільки, щоб змінна $y_к$ в явній формі визначалася з (3.119). За величиною похідної можна визначити вплив зміни тієї чи іншої дії на процес функціонування складної технічної системи.

У початковому процесі характеристики стану ТС можуть залежати не тільки від часу, але і від просторових координат. Із множини ТС із розподіленими параметрами можна виділити системи, параметри яких приводяться до зосереджених. Це такі системи, в яких достатньо знати значення вхідних та вихідних змінних у кінцевому числі фіксованих точок простору. Наприклад, лінійні елементи ТС із розподіленими параметрами структурно можуть бути представлені у вигляді багатомірної лінійної системи з зосередженими параметрами. На рис. 3.14 показано балку на двох опорах, яка під дією зовнішньої змінної сили P здійснює коливання відносно положення статичної рівноваги (приклад 3.14).

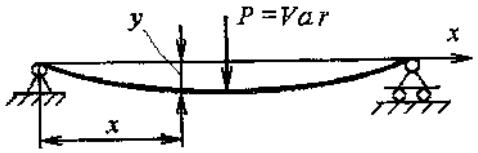

Рис. 3.14. Схема балки з розподіленими параметрами

У цій системі кожна точка балки здійснює своє переміщення y, яке залежить від положення координати x. Ця балка являє собою систему з розподіленими параметрами (масами) вздовж координати x, тобто тут маса балки і її прогин є координати довжини x: m(x) і y(x). Ці зміни: є характеристиками з розподіленими параметрами.

Розглянута система з розподіленими параметрами може бути замінена системою із зосередженими параметрами (рис. 3.15).

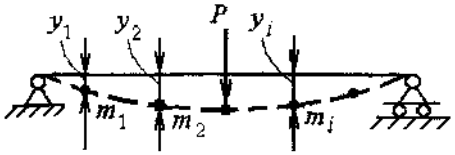

Рис. 3.15. Схема балки із зосередженими параметрами

У заміщеній схемі окремі елементи балки замінені масами m_i, m_2, ... і відповідними їм координатами y_1, y_2, ...При розгляді коливань балки під дією змінної сили Р умовою еквівалентності схем, показаних на рис. 3.14 та рис. 3.15, є рівність кінетичних енергій систем з розподіленими й зосередженими параметрами.

3.4.2. Аналіз моделей

Модель ТС, що досліджується, є формалізованим і спрощеним її описом. У ній ураховується тільки деяка підмножина Із множин ознак, які складають початковий опис системи. Вид моделі визначається не тільки природою реальної ТС, але й тими завданнями, для розв'язування яких будується модель, а також необхідною точністю їх розв'язування. Тому необхідні дослідження отриманої моделі з метою визначення області її найбільш ефективного застосування при розв'язуванні інженерних задач і встановлення меж зміни параметрів, у яких вона справедлива.

Ефективність використання математичних моделей для Дослідження ТС може бути показана за допомогою наступного прикладу.

3.4.3. Оцінка ідентичності моделі і технічної системи

Необхідна умова для переходу від дослідження технічної системи (ТС) до дослідження моделі і подальшого перенесення результатів на ТС — вимога адекватності моделі й ТС. Адекватність передбачає відтворення моделлю з необхідною повнотою всіх властивостей ТС, які суттєві для

даного дослідження. Оскільки будь-яка модель має характер певної проекції ТС, ніколи не можна говорити про абсолютну адекватність, при якій модель за всіма характеристиками відповідає оригіналу. Отже, оцінка ідентичності може спиратись тільки на оцінку відмінності моделі від оригіналу.

Поняття адекватності базується на математичних поняттях ізоморфізму і гомоморфізму. ТС, що досліджується, та її модель називаються Ізоморфними, якщо між ними існує взаємно-однозначна відповідність. Гомоморфізм, як і ізоморфізм, передбачає збереження в моделі всіх визначених у ТС властивостей та відношень. Однак тут вимога взаємно-однозначної відповідності замінюється вимогою однозначної відповідності моделі ТС, тоді як відповідність ТС моделі неоднозначна.

Ізоморфна модель включає всі риси, які теоретично притаманні оригіналу. При бажанні побудувати ізоморфну модель головна перешкода полягає у відсутності перетворення, яке встановлює необхідну взаємно-однозначну відповідність.

Гомоморфізм визначає таку форму зв'язку між двома подібними системами, коли однозначне лише в одну сторону перетворення дає змогу звести початкову систему до більш простої системи, яка гомоморфна початковій. Моделюванню ТС притаманний гомоморфізм.

Оцінити рівень ідентичності моделі і ТС можна за допомогою кількісних показників. Задача встановлення рівня ідентичності моделі і ТС може бути поставлена наступним чином: для відомої ТС будується її модель таким чином, щоб при подачі однакових вхідних дій на ТС і її модель вихідні сигнали мінімально відрізнялись один від одного. Рівень відхилень вихідних сигналів визначається кількісними показниками. Серед них можна виділити мінімум середнього квадрата похибки.

При побудові моделей досить часто попередньо невідомі ні ступінь впливу входів на виходи, ні форма залежності між окремими вхідними й вихідними змінними. При таких умовах оптимальну оцінку моделі ТС при використанні критерію мінімуму середнього квадрата похибки дозволяє дати

регресія виходу по відношенню до всіх входів, тобто умовне математичне сподівання виходу. Добавлення врахованих входів при побудові оцінки моделі збільшує дисперсію (квадрат відхилення) умовного математичного сподівання виходу.

У загальному випадку для p входів і одного виходу маємо

$$y^*(t) = M[y(t)/z_1(s_1), z_2(s_2), \ldots, z_p(s_p); s_k \in T_k], \qquad (3.126)$$

де $z_1(s_1), z_2(s_2), \ldots, z_p(s_p)$ - ураховані вхідні змінні, які змінюються на кінцевих інтервалах часу T_k, тобто $s_k \in T_k$, k=1, 2, ..., p; y*(t) - значення виходу моделі у фіксований момент часу t; y(t) - виміряне значення виходу ТС. Для одномірного випадку (один вхід і один вихід) вихідна змінна моделі визначається так:

$$y^*(t) = M[y(t)/z(s); s \in T]. \qquad (3.127)$$

Мірою близькості y(t) і y*(t) як випадкових функцій часу є дисперсія умовного математичного сподівання. Безумовна дисперсія виходу D[y(t)] може бути представлена у вигляді двох доданків [24]:

$$D[y(t)] = D\{M[y(t)/z(s); s \in T]\} + D[y(t)/z(s); s \in T]. \qquad (3.128)$$

Перший доданок в (3.128) характеризує ту частину загальної дисперсії, яка визначається змінами у врахованій змінній; другий — характеризує частину загальної дисперсії, яка відображає невраховані фактори.

Залежність (3.128) випливає із рівності

$$\begin{aligned} D(y) &= M[y - M(y)]^2 = \\ &= M[y - M(y/z) + M(y/z) - M(y)]^2 = \\ &= M[y - M(y/z)]^2 + M[M(y/z) - M(y)]^2 + \\ &+ 2M\{[y - M(y/z)][M(y/z) - M(y)]\} \end{aligned} \qquad (3.129)$$

Оскільки в рівності (3.129) останній доданок дорівнює нулю, то можна записати

$$D(y) = D(y/z) + D[M(y/z)],$$

тобто отримано залежність, яка відповідає (3.128).

За величину рівня ідентичності моделі і ТС приймають дисперсивну

міру

$$Q_{y/z}(t,T) = \frac{D\{M[y(t)/z(s);s \in T]\}}{D[y(t)]},$$

яку називають кореляційним відношенням.

За міру неідентичносгі моделі й ТС використовують

$$\bar{Q}_{y/z}(t,T) = \frac{D[y(t)/z(s);s \in T]}{D[y(t)]}.$$

Очевидно, що справедлива рівність Q + Q = 1.

Розглянемо деякі властивості дисперсивної міри.

1. Для детермінованої одномірної ТС умовна дисперсія виходу D(y/z) дорівнює нулю, тому дисперсія умовного математичного сподівання дорівнює безумовній дисперсії виходу, тобто

$$D[M(y/z)] = D[y(t)].$$

Таким чином, при всіх урахованих входах і при відсутності інших впливів на ТС математичне сподівання дорівнює самій вихідній величині.

У цьому випадку

$$Q_{y/z}(t,T) = 1;\ \bar{Q}_{y/z}(t,T) = 0. \qquad (3.130)$$

ТС, для яких справедливі рівності (3.130), називаються детермінованими, повністю визначеними або регулярними.

2. Якщо вихідна змінна y (t) ніяк не зв'язана з входом z(s);s∈T, то (оскільки в цьому випадку M(y/z)=M(y)) з виразу для дисперсії умовного математичного сподівання D[M(y/z)] = M[M(y/z)–M(y)]^2 випливає, що D[M(y/z)]= 0. Тому справедлива рівність

$$D(y/z) = D[y(t)].$$

У цьому випадку дисперсивна міра

$$Q_{y/z}(t,T) = 0 \qquad (3.131)$$

при умові, що D[y(t)] ≠ 0. ТС, для яких можливе виконання рівності (3.131), називають невизмаченими або нерегулярними.

4. АНАЛІЗ І СИНТЕЗ ТЕХНІЧНИХ СИСТЕМ

Одними з найважливіших задач дослідження технічних систем (ТС) є задачі аналізу й синтезу. Аналіз — це метод вивчення системи який базується на поділі її на частини, що вивчаються окремо шляхом абстрагування від впливу інших частин. При цьому задачі дослідження системи суттєво спрощуються. В процесі аналізу за відомою структурою і параметрами ТС вивчається її поведінка, тобто досліджуються властивості системи та її характеристики.

Синтез — це метод вивчення системи, який грунтується на розгляді її частин у взаємодії між собою, їх взаємному впливі і зв'язках. Синтез дає повне уявлення про об'єкт дослідження як цілісну систему. При цьому задачі синтезу системи порівняно із задачами аналізу значно ускладнюються. Процес синтезу ТС полягає в знаходженні її структури і визначенні параметрів за заданими властивостями.

Задачі аналізу та синтезу ТС взаємозворотні і, як правило, розв'язуються спільно. Так задачі синтезу як більш складні найчастіше всього розв'язуються з використанням результатів розв'язування задач аналізу.

4.1. Аналіз технічних систем

4.1.1. Задачі аналізу

Як уже відомо, аналіз — це процес визначення або дослідження властивостей, які притаманні технічній системі. Нехай відомі функції і характеристики елементів, що входять до складу системи, і визначена її структура. Необхідно визначити функції і характеристики, які притаманні системі як сукупності елементів.

Аналіз систем з метою визначення й оцінки їх якісних і кількісних властивостей є однією з найважливіших задач теорії технічних систем.

Аналіз дає змогу оцінити властивості різних класів ТС, їх структур, стратегій керування системами; характеристики як окремих елементів, так і їх сукупності.

Показники, що характеризують властивості ТС, можуть бути визначені одним із двох способів: 1) шляхом обробки результатів натурного експерименту; 2) в результаті фізичного або математичного моделювання процесів функціонування системи.

Вивчення системи в натурних умовах практично доцільно тільки при виконанні наступних умов:

• система може функціонувати в режимах, які дають можливість досягти цілі експерименту;

• є можливість фіксації всієї необхідної інформації без суттєвих витрат на датчики і накопичувачі інформації;

• фіксація та статистична обробка отриманої інформації в реальному масштабі часу задовольняє поставленим термінам експерименту;

• зміна режиму функціонування системи не приводить до аварії.

Оскільки в більшості практичних випадків перераховані умови не виконуються, то найбільш ефективним засобом аналізу складних ТС є їх математичне моделювання, яке детально описано в попередньому розділі.

Задача аналізу ТС включає три етапи.

На першому етапі необхідно виявити причинно-наслідкові зв'язки, які притаманні системі, яка аналізується, й побудувати ЇЇ концептуальну (причинно-наслідкову) модель, що розкриває суть процесів, які проходять в системі. При побудові концептуальної моделі встановлюється наявність залежності між характеристиками процесу, що цікавлять дослідника, і параметрами системи. Ці параметри повинні бути закладені в модель системи.

На другому етапі на базі прийнятої концептуальної або динамічної моделі будується математична модель, яка виявляє кількісні співвідношення між характеристиками процесу та параметрами системи. Така модель може

бути задана, наприклад, у вигляді функціональної залежності Y= Ф (X, U), де Y - множина вихідних характеристик системи; X - множина параметрів, що враховуються концептуальною або динамічною моделлю; U - множина вхідних дій на систему. Кількісні співвідношення конкретизують причинно-наслідкові зв'язки І тим самим повністю визначають модель системи. Дослідження залежностей Y= Ф (X, U) дає змогу виявити властивості системи, граничні і екстремальні значення характеристик, взаємні зв'язки між ними.

Оскільки побудова моделі здійснюється формальними методами, то виникає необхідність перевірки достовірності моделі та отриманих на її основі теоретичних результатів, що і здійснюється на третьому етапі розв'язування задачі аналізу. Перевірка достовірності проводиться шляхом зіставлення отриманих із моделі залежностей з експериментальними даними або даними, які отримані іншими методами аналізу.

Результатом аналізу є моделі процесів, що проходять в системах, і закономірності, які притаманні процесам та системам. Моделі розкривають причинно-наслідкову природу процесів і встановлюють залежності між їх характеристиками й параметрами системи Саме в цьому полягає пізнавальна цінність аналізу. Прикладна цінність аналізу зумовлена використанням результатів аналізу для постановки задач синтезу, які виникають при проектуванні технічних систем.

Дослідження складних ТС починається з аналізу властивостей алгоритмів, різних стратегій керування процесами, способів організації систем у цілому. При цьому будуються і досліджуються моделі процесів, що проходять у системах, які реалізують різні класи прикладних задач на основі різних структур та стратегій керування процесами. Результати аналізу сприяють розумінню суті процесів, що проходять у складних ТС.

4.1.2. Формалізація і постановка задачі аналізу технічних систем

Автоматизовані (машинні) методи аналізу вимагають особливої ретельності в описанні задач, які стоять перед інженером-дослідником і інженером-проектувальником конкретної ТС. В цьому випадку процес аналізу повинен бути по можливості строго формалізованим. Це означає, що він повинен підпорядковуватись жорстким закономірностям, які не допускають довільної інтерпретації. Структура й характер вхідної інформації та інформації, яку отримано в результаті процесу, повинні бути жорстко фіксованими. В таких випадках кажуть про алгоритмізацію процесу аналізу ТС. Під алгоритмізацією розуміють точну послідовність операцій аналізу, яка задає обчислювальний процес, що починається з довільних вхідних даних, і направлена на отримання повністю визначеного цими даними результату.

У певних випадках для того, щоб зробити процес аналізу технічної системи більш ефективним, виникає потреба надати людині (інженеру) можливість у необхідних та строго визначених місцях цього процесу втручатись у його хід, тобто розформалізувати його. Це не зменшує важливості жорсткої формалізації процесу аналізу, а робить його більш гнучким.

Розглянемо тепер довільну технічну систему, яка є частиною навколишнього світу. Останній має вплив на ТС і характеризується параметрами входу u_1, u_2, ..., u_m, які можуть бути позначені вектором входу $U = \{u_1, u_2,..., u_m\}$. Якщо вхідні характеристики залежать від часу t, то $U = U(t)$. Вхідні характеристики можуть бути детермінованими або випадковими. Класичним і досить поширеним прикладом детермінованого впливу на систему є синусоїдальні коливання виду $U(t) = \{A_1 \sin(\omega_1 t + \varphi_i\},...,A_m \sin(\omega_m t + \varphi_m)\}$, де $A_1...,A_m$; ω_1, ..., ω_m; φ_1, ..., φ_m – амплітуди, частоти й початкові фази коливань. Випадкові дії інколи є корисними, а інколи мають негативний вплив на систему. Загальним для цих дій є те, що вони випадкові

(стохастичні), та їх поведінку не можна достовірно передбачити завчасно.

До цього часу ми говорили про вплив на систему із зовні, але метою будь-якої системи є створення певного технічного ефекту, що діє на зовнішній світ. Цей ефект, як і вхідні дії, можна описати набором певних функцій часу, які у векторній формі мають вигляд $Y = \{y_1, y_2,\ldots,y_n\}$. Ці функції називають відгуком або виходом системи. Виходом ТС може бути струм, напруга, переміщення, прискорення й ін.

Виходячи з розглянутих вхідних та вихідних характеристик, можна сказати, що задачею технічної системи є перетворення компонентів вектора U, які не заважають системі, в компоненти вектора Y, які дають корисний ефект (рис. 1.3).

Щоб охарактеризувати задачу аналізу, необхідно мати ще одне, й досить важливе поняття — це опис технічної системи, яке може бути представлене у вигляді вектор-функції часу $X(t) = \{x_1(t), x_2(t), ..., x_1(t)\}$. Опис системи повинен бути настільки повним і точним, щоб задача аналізу могла бути розв'язана. В різних галузях техніки використовують різні форми опису: диференціальні рівняння руху механічної системи; операторні рівняння системи керування; алгебраїчні рівняння типу законів Ома і Кірхгофа в електротехніці; креслення з числовими даними машин та механізмів тощо.

Тепер ми можемо описати задачу аналізу ТС в одній із самих відомих її постановок: задані вхідні дії U(t) на систему і її опис X(t), необхідно знайти відгук або виходи системи Y(t).

Фактично інженер або дослідник, який аналізує ТС, розв'язує, як правило, більш складну задачу. Справа в тому, що визначення відгуку Y(t) є часто не єдиною і більше того не завжди головною задачею дослідника. У зв'язку з цим виникає питання, якими параметрами й функціями характеризується ТС. Для того, щоб відповісти на це питання, введемо систему понять, які пов'язані з результуючою інформацією аналізу ТС. Ця система понять включає в себе: 1) набір функцій та чисел, що описують поведінку (функціонування, динаміку) системи; 2) набір функцій і чисел, що

відповідають характеристикам системи; 3) набір функцій і чисел, що описують властивості системи. При такому підході функції y_1 y_2,...,y_n являють собою окремі випадки функцій і чисел, що складають перший елемент тріади.

Спираючись на введені поняття, розглянемо задачу аналізу в найбільш повному вигляді: задані вхідні (корисні і зайві) дії U(t) на систему, а також її опис X(t), наприклад, у формі диференціальних рівнянь $\dot{X}(t) = f(t,U(t))$; необхідно знайти набір функцій та чисел, що описують поведінку, характеристики й властивості ТС.

При традиційному ("ручному") аналізі знаходження цих функцій і чисел виконується аналітичне, але це не завжди можливо. Наприклад, у випадку, коли диференціальні рівняння стану системи не вдається аналітичне проінтегрувати. В цьому випадку необхідно використовувати чисельні методи інтегрування й обчислювальні засоби (ЕОМ). Крім того, в процесі аналізу технічної системи ЕОМ може виконувати допоміжні функції: будувати графіки функцій за раніше знайденим аналітичним виразом, перевіряти виконання певних нерівностей, зберігати і накопичувати різну Інформацію і т. д. При машинному аналізі розрахунок чисел та функцій, що складають тріаду, повністю виконує ЕОМ. За людиною залишаються високоінтелектуальні аспекти цього процесу: вибір методу аналізу, перехід до іншого методу, прийняття рішення про зміну формулювання задачі аналізу, співставлення результатів, отриманих різними способами, зіставлення результатів машинного аналізу з відомими експериментальними даними тощо.

Процес аналізу ТС залежить від того, детерміновані чи випадкові компоненти вхідної дії. В першому випадку вхідні дії задаються звичайними функціями часу U(t). Це можливо зробити, як відомо, за допомогою аналітичного виразу, графіка, таблиці значень функції або алгоритму її розрахунку за допомогою ЕОМ. Якщо компонент u(t) вектор-функції U(t) випадковий, необхідно користуватися методами опису випадкових функцій.

Це роблять, описуючи звичайним шляхом деякі спеціально введені детерміновані функції, наприклад, функції густини ймовірності або кореляційні функції.

Коли U залежить, крім t, ще і від просторових координат, то ми маємо справу з описом функції дії від багатьох змінних. Це ускладнює задання функції, однак принципи опису залишаються такий ми ж, як і для випадку з однією змінною.

4.1.3. Технологія аналізу технічної системи

Як буде показано пізніше, задача аналізу значно простіша від задачі синтезу ТС. Пояснюється це в основному тим, що задача аналізу легше формалізується. Поведінку обчислювача (людини або ЕОМ) тут простіше уявити у вигляді жорсткої, завчасно заданої послідовності операцій, що не допускає, як і самі операції, ніякої зміни. Складові операції аналізу можуть бути достатньо складними, а їх кількість значною і повторюватися в розрахунках декілька разів. Але це не змінює суті справи, а лише показує необхідність використання ЕОМ для аналізу ТС.

Структура машинного аналізу ТС може бути у загальних рисах описана в декілька етапів наступним чином.

На першому етапі дослідник уводить у машину опис ТС. Тут важливими є два моменти. Мова, на якій зроблено опис системи, повинна бути легко доступною машині. Найкраще, коли опис системи зводиться до послідовності буквених, цифрових і деяких спеціальних символів, завчасно обумовлених. З іншого боку, мова повинна бути простою, яку в змозі освоїти дослідник або інженер, що не має спеціальної підготовки, без значних витрат часу і коштів.

Другий етап. Обчислювальна машина перетворює опис системи у форму, зручну для подальших операцій аналізу. Досить часто трапляється, що основною операцією аналізу ТС є розв'язання системи алгебраїчних

рівнянь або інтегрування системи диференціальних рівнянь. Тому необхідно ввести в машину, наприклад, систему диференціальних рівнянь, що описують поведінку ТС. Коли ЕОМ розв'язує чисто математичну (або, краще сказати, сформульовану як чисто математичну) задачу, то, звичайно, так і поступають. Однак якщо аналізується ТС, і особливо якщо вона складна, то подібний підхід не досить раціональний.

Складання опису мовою алгебраїчних, диференціальних або Інших рівнянь досить часто зіставлюване за трудомісткістю з нашими операціями процесу аналізу, наприклад, із самим розв'язуванням рівнянь. Тому бажано автоматизувати процедуру складання і формування рівнянь, що описують поведінку ТС. Формування рівнянь належить до тих операцій, які допускають жорстку Формалізацію. Тому машина, як правило, краще справляється з цією операцією, ніж людина.

Тут ми зустрічаємося з випадками взаємодії людини та ЕОМ. Тому корисно замітити, що в ідеалі така взаємодія повинна підпорядковуватись простому принципу: те, що піддається формалізації, — машині, що ні (інтелектуальне, інтуїтивне, пов'язане з накопиченням досвіду, адаптації, з неформальними розв'язками) -людині. На розглянутому етапі аналізу вдало використовується цей принцип.

Закінчується процес перетворення опису системи розробкою його стандартної універсальної форми. Після цього, щоб сформулювати опис, машині необхідно розрахувати певні набори чисел і функцій, якими відрізняється опис однієї конкретної системи від іншої у формі, яка прийнята за стандартну.

Третій етап. Мета аналізу полягає в розрахунку певної сукупності чисел і функцій, які описують поведінку, характеристики і властивості ТС. Ці числа та функції можуть з'явитися тільки як результат переробки машиною опису системи, який розроблено на другому етапі аналізу. Ця переробка, тобто власне розрахунки, і становить суть третього етапу аналізу.

Частіше за все на третьому етапі проводиться розв'язування тих рівнянь,

які було сформульовано на другому етапі. В нашому прикладі з лінійними рівняннями третій етап являє собою розрахунок вектора X. Якщо опис системи було сформульовано мовою диференціальних рівнянь, то третій етап полягає у їх інтегруванні.

Таким чином, третій етап аналізу здається чисто математичним за своїм характером. Однак це не зовсім так. Будь-яку обчислювальну задачу, в даному випадку знаходження коренів системи (4.1), або інтегрування диференціальних рівнянь, можна розв'язати по-різному, вибравши для цього певний метод розв'язування. Далі обчислювальний метод ще не визначає однозначно алгоритму розв'язування, тобто жорсткого набору дій в усіх деталях. Нехай, наприклад, певний спосіб розв'язування задачі містить деякі подібні A у формулі (4.2). Виявляється, що цю операцію можна здійснити за допомогою різних алгоритмів. Виконуючи ті операції, які вимагає третій етап аналізу, ми повинні фіксувати обчислювальний метод і алгоритм.

Щоб обчислювальний процес, який складає суть третього етапу, міг бути фактично реалізованим, необхідно вибрати обчислювальний метод, який реалізує його алгоритм і чисельні параметри алгоритму. Тут ми знову повертаємось до проблеми взаємодії людини й ЕОМ у процесі аналізу. Всі ці проблеми вибору можна розв'язати трьома способами: 1) автор програми робить вибір обчислювального методу завчасно для всіх доступних їй задач; 2) ЕОМ, аналізуючи процес розв'язування задачі та отримані результати, згідно із завчасно введеними в неї правилами, змінює процес розв'язування, про який ішла мова в першому пункті; 3) нові розв'язки приймаються не ЕОМ, а інженером під час обчислювального процесу залежно від того, як розвивається цей процес.

Усі ці проблеми вибору тісно пов'язані не тільки з математичними аспектами задачі аналізу, але і з фізичними та технічними особливостями конкретної системи, яка підлягає аналізу. Наприклад, один і той же метод інтегрування диференціальних рівнянь може бути досить вдалим та зовсім неефективним залежно від того, яка система. Врахування особливостей ТС

дуже важливе при перегляді прийнятих розв'язків. Невдачу або удачу зробленого вибору кваліфікований інженер-дослідник, звичайно, може зв'язати не тільки і не стільки з формально-математичними особливостями опису системи, скільки з її фізичними й технічними властивостями та особливостями (інколи зовсім незрозумілими математику-прикладникові). Тому фізична інтерпретація тих результатів, до яких привів вибір методу, алгоритму і параметрів алгоритму фізична та інженерна інтуїція інженера-дослідника виявляються досить важливими й визначальними при розв'язуванні задач аналізу ТС.

Однак отримати бажаний результат аналізу технічної системи можна тільки в тому випадку, коли забезпечені можливості зручного діалогу між людиною та ЕОМ. Забезпечення як програмних, так і апаратних засобів для такого діалогу є однією з основних цілей розробників програм аналізу та синтезу ТС. Саме такий діалоговий режим обчислювального процесу дає змогу інженерові-досліднику активно втручатися в розрахунки й вирішувати проблеми вибору методів і алгоритмів обчислень. Добрі програми аналізу ТС повинні забезпечити спілкування між людиною та машиною і робити його по можливості зручним, тобто вимагати незначної кількості вказівок з контролем помилок та невідповідностей у цих вказівках.

Четвертий етап аналізу технічної системи пов'язаний з обробкою результатів, отриманих у рамках процесу діалогу між людиною й ЕОМ: побудова графічних матеріалів за допомогою графічних пристроїв ЕОМ, вивід інформації на монітор, друкування таблиць чисел і текстових документів, передача результатів у бази даних, де накопичуються результати різних досліджень, об'єднаних спільністю типів технічних систем, задач аналізу тощо. Одні і ті ж результати, що отримані на третьому етапі, можуть за вимогою дослідника приймати різні форми в процесі обробки. Наприклад, сукупності (масиви) чисел, що відповідають набору певних кривих, можуть бути перетворені в таблиці функцій, графіки, таблиці чисел, гістограми (оцінки густини ймовірностей випадкових величин) і т.д. У Свою чергу,

гістограми можуть бути надруковані й виведені на монітор у вигляді таблиць, графіків тощо. Вказівки про характер обробки результатів вихідної інформації можуть Видозмінюватися в процесі аналізу технічної системи.

Етап аналізу, пов'язаний з обробкою вихідної інформації, не можна недооцінювати, оскільки вдало знайдена форма представлення результатів дає можливість досліднику знайти нові ідеї, які дають змогу більш повно розкрити фізичні й технічні процеси в системі, знайти інші підходи до аналізу, змінити початковий опис системи і т. д.

4.1.4. Структура процесу аналізу технічної системи

У багатьох випадках до інформації, яка міститься в початковому описі технічної системи (ТС), додається інформація іншого роду, котру називають апріорною. Це — вся та інформація, яку інженер-дослідник повинен обробити і ввести в ЕОМ до початку обчислень, щоб зробити розв'язок задачі аналізу можливим при доступних ресурсах ЕОМ. Під ресурсами ЕОМ розуміють сукупність таких величин, як машинний час, об'єм використаної пам'яті й ін.

Збирання та оброблення апріорної інформації вимагають певних витрат ресурсів — людських і машинних. Тому дослідник, організуючи й плануючи процес машинного аналізу, майже завжди розв'язує таку задачу; збирання та оброблення апріорної інформації або витрати ресурсів під час розрахунку на ЕОМ.

Під час аналізу виникають питання повноти і точності опису ТС. До початку аналізу не вдається знайти відповіді на поставлені питання, оскільки багато умов та параметрів, що в них входять, можна перевірити лише після завершення процесу аналізу. Такі ж питання виникають і про повноту апріорної інформації. Все це є характерним для процесу аналізу складних ТС, де неможливо завчасно передбачити, як буде проходити процес, з якими кількісними похибками він здійснюється, який характер будуть мати

отримані числа і функції. Тому доцільно будувати цей процес як послідовність певних проб, даючи можливість інженерові-досліднику повертатися до вже пройдених етапів процесу, зі зміною початкової інформації, методів та алгоритмів розрахунку і т. д.

З урахуванням цих застережень процедуру аналізу ТС можна здійснювати в такій послідовності.

1. Уведення початкового опису системи. Визначення функцій і чисел, які необхідно отримати після закінчення аналізу.

2. Перетворення початкового опису у форму, яка зручна для ЕОМ (формування опису).

3. Збір, обробка та введення апріорної інформації.

4. Власне аналіз (обчислення на ЕОМ, які мають на меті отримання функцій і чисел, що вказані в першому пункті).

5. Перевірка умови, яку можна сформулювати наступним чином: чи достатньо інформації, яка міститься в описі системи, й апріорної інформації? Якшо достатньо, то процедура аналізу продовжується, а якщо ні, то необхідно доповнити недостатню інформацію і повернутися до пунктів 1 або 3 цієї процедури.

6. Обчислення та виведення результатів аналізу на зовнішні Носії інформації,

7. Перевірка умови: чи всі цілі аналізу досягнуті і чи потрібно доповнювати їх новими.

Аналіз системи закінчується, а якщо потрібні — повертаємось на початок процедури, вказавши новий набір необхідних функцій чисел тощо.

Побудована таким чином процедура аналізу ТС являє собою циклічну структуру, в якій певні операції можуть виконуватися декілька разів. Тут необхідно відзначити, що будь-які розв'язки, які можуть бути прийняті інженером-дослідником у процесі аналізу, носять неформальний характер, тому роль останнього в цьому процесі є визначальною.

4.1.5. Формування опису технічної системи

Вимоги до форми опису ТС із боку інженера-дослідника та ЕОМ, які диктуються найкращою організацією обчислювального процесу, суттєво різні. Інженерові-досліднику важлива перш за все простота опису, тобто форма опису повинна бути такою, щоб його могла скласти мало кваліфікована людина. В той же час обчислювальний процес може бути здійснений лише тоді, коли початковий опис буде перетворений у форму, яка є зручною для ЕОМ.

4.1.6. Апріорна інформація

Апріорною інформацією називають інформацію, яку необхідно зібрати, опрацювати і ввести в ЕОМ, щоб можна було розв'язати задачу аналізу ТС при допустимих машинних ресурсах або зменшити вимоги до них.

Розглянемо три випадки, які показано на рис 4.2. Для простоти ми обмежимось областями на площині, тобто вважаємо, що вектори X двомірні.

На рис. 4.2 кружечками відмічено ті точки області G, які дають періодичні режими руху технічній системі.

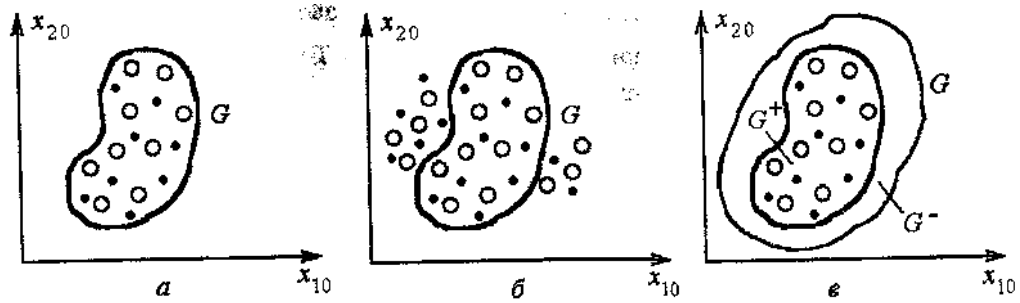

Рис. 4.2. Область G початкових умов, що забезпечують існування заданих властивостей (періодичних режимів руху) технічної системи

Ситуація, що показана на рис. 4,2, а, відповідає ідеальному випадку: область G складається тільки з точок, які з максимальною точністю і

повнотою описують апріорну Інформацію. На рис. 4.2, б область G складається з точок, які не повністю описують апріорну інформацію, бо за межами області G є точки, які задовольняють початкові умови для циклічних режимів руху технічної системи. Область G повинна була б бути більш повною, ніж на рис. 4.2, б. Однак нам це невідомо, і, можливо, повна межа області визначиться лише після розрахунків на ЕОМ. Третій можливий випадок показано на рис 4.2, в. Тут область G охоплює не тільки бажані точки (вони утворюють область G^{+}), але і зайві (область G^{-}). Очевидно, почавши розрахунки з точок, котрі належать області G^{-}, ми не отримаємо бажаного результату. В цьому випадку наша апріорна інформація неточна.

З розглянутого прикладу можна зробити висновок, що існують два небажаних випадки опису апріорної інформації на стадії попереднього дослідження системи: неповнота (рис. 4.2, б) і неточність (рис. 4.2, в). Кількісна оцінка цих факторів можлива лише ймовірнісними методами. Наприклад, можна оцінити ймовірність ситуації, яка подібна тій, що показана на рис. 4.2, 6, або оцінити статистичні характеристики випадкових величин, що характеризують область G^{-}.

Жорсткі рекомендації до вибору апріорної інформації в тому чи іншому вигляді (рис. 4.2) дати практично неможливо. Для випадку, показаному на рис. 4.2, а, необхідні значні витрати ресурсів на підготовку апріорної інформації, але при цьому значно зменшуються витрати на обчислювальні операції. У випадках (рис.4,2, б, в), навпаки, зменшуються витрати на підготовку апріорної інформації, зате збільшуються витрати на обчислювальні операції. При цьому співвідношення між одними і другими витратами значною мірою залежать від точності визначення області G, яка може бути звуженою або розширеною.

4.1.7. Приклад машинного аналізу технічної системи

Розглянемо структуру процесу машинного аналізу на прикладі розрахунку періодичних коливань технічної системи. Припустимо, що розв'язок системи (4.7) при вибраних нами початкових умовах належить області G_x і є неперіодичним. У процесі діалогу з ЕОМ дослідник у завчасно визначених ним точках осі t буде отримувати інформацію про розв'язок X(t) та його періодичність. При наявності цієї інформації неперіодичність процесу руху механічної системи можлива в двох випадках: 1) розв'язок в дійсності неперіодичний; 2) розв'язок періодичний, але розрахунок перехідного процесу ще не закінчено. Вияснити причину неперіодичності процесу дослідник може декількома шляхами.

1. Спинити процес обчислень через деякий час і тим самим внести певний вклад у задачу аналізу — визначити властивості системи. Прийняттю такого рішення може сприяти апріорна інформація іншого характеру, ніж та, яку пов'язували з областю G_x. Такою інформацією може бути час установлення періодичного режиму руху механічної системи t_B. Якщо за апріорними даними час установлення періодичності руху повинен бути певної величини, а розрахунковий відрізок процесу складає до моменту прийняття рішення значно більшу величину часу, то рішення про спинення обчислювального процесу є виправданим. Однак при цьому задача аналізу з розрахунку періодичності руху механічної системи залишається невиконаною.

2. Продовжувати процес обчислень, щоб з'ясувати, чи встановиться в подальшому періодичний режим руху. При цьому також можна використати допоміжну апріорну інформацію.

3. Частково спинити процес обчислень і перевірити достатність описової й апріорної інформації. Може бути, що апріорної інформації достатньо, а описової — недостатньо або навпаки. Може статися так, що необхідно уточнити апріорну інформацію, наприклад, величину t_B. Ця величина може

виявитися випадковою. Ми вважаємо, що технічна система і зовнішні дії на неї детерміновані, тому й t_B повинно бути детермінованим числом. Однак, якщо система складна та інформація про t_B апріорна, то визначити величину t_B з достатньою точністю неможливо. Використовуючи будь-який метод оцінки, вдається визначити лише діапазон, у якому з певною ймовірністю повинна знаходитися величина t_B.

4.2. Синтез технічних систем

4.2.1. Суть задачі синтезу технічної системи

Постановка задачі синтезу певною мірою зворотна постановці задачі аналізу. В процесі синтезу задаються описи вхідних дій і описи поведінки, характеристики й властивості майбутньої системи. Досить часто в описи поведінки системи входять описи виходів. Ці описи або їх частина в аналізі не задавались, а шукалися. В задачі синтезу необхідно знайти опис самої системи та її стани, в той час як у задачі аналізу опис системи заданий. Таким чином, розв'язування задачі синтезу являє собою процес перетворення одних описів в інші. Ця задача аналогічна задачі аналізу, тільки вхідні і вихідні описи тут інші Й сам процес перетворень описів також інший. Саме ці принципові різниці роблять процес синтезу більш творчим, де головну роль відіграє кваліфікований спеціаліст, якого не може замінити ніяка сукупність обчислювальних засобів. Однак нерозумно завантажувати спеціаліста громіздкою одноманітною інформацією, яка не вимагає творчих здібностей та інтуїції. Тому в процесі синтезу технічних систем бажано організувати взаємодію людини й ЕОМ.

У процесі синтезу технічних систем центральне місце займають проблеми вибору структури системи і базисних елементів. Ці проблеми в багатьох задачах дуже погано формалізуються та алгоритмізуються. Саме тут повинна проявлятися творча сила людського інтелекту, його вміння

користуватися інтуїцією, досвідом розв'язування подібних або суттєво інших задач і т. д.

Одним із шляхів синтезу технічних систем є вибір певної структури й базисних елементів і на основі розв'язування задачі аналізу здійснення зондування параметрів системи та вибір таких з них, які задовольняють бажані властивості системи. Якщо аналіз направлений, то його можна будувати таким чином, щоб наблизитися до бажаних властивостей системи. Так чи інакше нам удається встановити залежність між характеристиками властивостей системи і параметрами її базисних елементів.

На перший погляд здається, що такий підхід досить простий. Однак в дійсності нам не зовсім зрозуміло, яку саме систему необхідно аналізувати. Система тільки створюється, і її опис нам необхідно знайти, тоді як для задачі аналізу він повинен бути відомим. Так ми стикаємося з основним протиріччям задачі синтезу, яке принципово усунути неможливо. Створення системи вимагає інформації про її поведінку, характеристики та властивості, а саму систему ще треба відшукати. Раніше, ніж опис системи буде знайдено, цю інформацію неможливо отримати, а не знаючи її, неможливо створити систему.

Основне протиріччя процесу синтезу пов'язане й з іншими, також достатньо суттєвими причинами, їх можна охарактеризувати таким чином: створити систему вдається лише тоді, коли ми знаємо, яку систему ми хочемо створити. Однак це знання приходить лише в процесі знаходження необхідної нам системи і, по суті, багато питань залишаються неясними і після того, як опис системи знайдено, а сама вона реалізована. Бувають випадки, коли постановка задачі синтезу виявляється лише після досить довгої експлуатації системи, тобто коли сама проблема синтезу, можливо, вже не викликає інтересу. Для підтвердження цієї думки розглянемо приклад.

Досвід створення складних технічних систем показує, що на початку процесу синтезу неможливо формалізувати всі умови, щоб вважати систему задовільною або, тим більше, найкращою з можливих. Визначеність у заданні

вимог приходить лише з розумінням поведінки, характеристик і властивостей системи — тієї самої системи, опис якої невідомий та який необхідно знайти. Звичайно, ці протиріччя майже не проявляються у випадку нескладних систем, поведінка, характеристики й властивості яких по суті легко передбачувані, хоча б якісно.

Зрозуміло, якщо процес синтезу завершено, то описане вище протиріччя повинно бути рано чи пізно усунутим. Знімається воно одноразовим або багаторазовим звертанням до процесу аналізу та оцінкою його результатів. Таким чином, аналіз виступає основним засобом для зняття протиріччя, що виникає в процесі синтезу технічних систем.

Принципові труднощі процесу синтезу — незнання того, що можливе і що неможливе в створюваній системі, і невміння формалізувати побажання проектувальника — можуть бути зняті, якщо проаналізувати деякий варіант системи. Виходячи з цього, можна запропонувати таку структуру процесу:

1. Розробити варіант опису технічної системи, який задовольняє початкові вимоги синтезу.

2. Провести ґрунтовний аналіз запропонованої системи. На цьому етапі розробити варіант опису технічної системи, який задовольняє початкові вимоги синтезу.

3. Провести ґрунтовний аналіз запропонованої системи. На цьому етапі бажано використати ЕОМ.

4. Оцінити переваги запропонованої системи на основі отриманої інформації про її поведінку, характеристики і властивості.

Якщо система задовольняє необхідні вимоги або, тим більше, є найкращою із усіх можливих, то процес синтезу необхідно закінчити, а якщо — ні, то повернутися до першого пункту. Така структура синтезу дає змогу людині й ЕОМ робити те, що у кожного з них виходить найкраще. Пропонування ідей, формування гіпотез, оцінка — все, що вимагає неформального, творчого підходу, залишається за людиною. Все, що краще алгоритмізується, в основному виконується ЕОМ. Але справа не тільки в

цьому. Кожний вдалий крок в описаній вище ітеративній, циклічній процедурі дає можливість що-небудь нове зрозуміти в системі, яка створюється: поступово визначаються межі можливого і неможливого, виявляються недооцінені або непомічені небезпечності, формалізуються цілі синтезу й ін. Таким чином, крок за кроком усуваються перешкоди, знімаються протиріччя.

Описана структура процесу синтезу є лише основою складного, розгалуженого процесу. Наприклад, задання опису системи вимагає задання структури, базису елементів і параметрів. Чи дає виявлення всіх цих компонентів можливість запропонувати варіант опису системи? Здебільшого — ні. В усякому разі, визначення параметрів системи людина виконує гірше, ніж на ЕОМ. Розрахунок параметрів часто вдається представити у вигляді стрункого алгоритму, який допускає ефективне використання ЕОМ. Тоді за людиною залишається творча робота, а отримані під час цієї роботи результати опису системи доповнюються чисельною інформацією від ЕОМ.

Розглянемо варіант, коли необхідно повертатися до першого етапу із заміною нового варіанта структури системи та її базису. Тут необхідно враховувати, що всі накопичені в процесі синтезу знання — виконання процедури аналізу (етап 2), оцінка отриманого рішення (етап 3), — все це впливає на формування ідеї про нову структуру або про новий базис елементів. Переглянувши декілька варіантів, наприклад, поперечних перерізів елементів колони (рис. 4.1), з'ясувавши, які з поставлених умов були виконані, а які ні, зрозумівши, якою він хоче бачити систему, конструктор пропонує такі нові описи системи, що після розрахунку їх параметрів можна прийти до висновку або про її прийнятність (кращого варіанта йому не знайти), або неможливість розв'язати задачу синтезу системи.

4.2.2. Про зміну постановки задачі синтезу

Як було показано раніше, синтез нерозривно пов'язаний із розумінням того, як веде або повинна вести себе система, які її властивості, характеристики, параметри і т. д. Приріст відповідної інформації не може не змінити постановки задачі синтезу. Якщо розглянути приклад 4.5 із паразитними коливаннями механічної системи, то можна говорити про дефект початкової постановки: вона не врахувала важливої вимоги, яка б забороняла системі проявляти коливальні властивості поза робочим діапазоном частот. Після того, як цей недолік системи був проаналізований конструктором, він, природно, ввів таку заборону, і постановка задачі змінилась.

Однак постановка задачі може змінюватися більш суттєво.

4.2.3. Способи оцінки технічних систем

Розглянемо два основних способи оцінки технічних систем, які визначають і постановку задачі, і значною мірою структуру алгоритму синтезу.

Перший спосіб пов'язаний з розрахунком певного набору чисел та функцій. При цьому кожній із систем відповідає одна й та ж множина чисел і функцій, хто б і коли їх не розраховував та незалежно від умов розрахунків.

В останньому прикладі набір чисел і функцій складався з одного елемента δ.

Після того, як числа і функції визначені, перевіряється виконання системи нерівностей або інших подібних співвідношень. Така перевірка завжди та однозначно приводить або до придатності системи, або до її непридатності. В розглянутому прикладі 4.6 достатньо було з'ясувати справедливість однієї нерівності (4.8). У більш складних випадках може виявитися необхідність перевірки, чи належить якась характеристика

створюваної системи завчасно вказаній області. Наприклад, розробник може вимагати, щоб характеристика v(t) всіх придатних варіантів системи містилась у завчасно вказаній області (рис. 4.4).

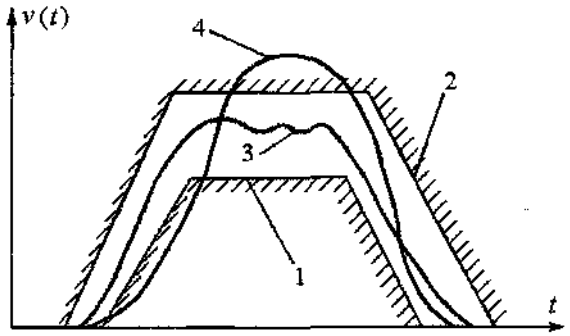

Рис. 4.4. Задання області, до якої повинні належати характеристики всіх систем, що синтезуються

Тоді оцінка придатності системи зводиться до поточкової (для кожної абсциси t, розміщеної між t_0 і t_1) перевірки приналежності числа v(t) відповідному відрізку, який обмежений ламаними 1 і 2 (рис. 4.4). Виходячи з такого способу оцінки технічних систем, характеристика 3 задовольняє вказані вимоги, а характеристика 4 — ні.

Такий жорстко формалізований, однозначний, що не допускає ніяких неформальних моментів, підхід називають кардиналістським підходом до синтезу технічних систем [26].

Другий спосіб. Альтернативним до розглянутого способу оцінки технічних систем є ординалістська трактовка синтезу [26]. Тут передбачається, що неможливо (недоцільно) використовувати для оцінки технічних систем тільки набір чисел і функцій або на основі цього набору неможливо (недоцільно) завжди й однозначно давати висновок про придатність системи.

До цих пір ми вважали, що мета синтезу полягає в знаходженні опису придатної (задовільної) системи. Такий підхід відповідає неоптимальному синтезу технічних систем.

Однак у багатьох випадках розробник хоче досягти максимальної мети

— йому потрібна не задовільна, а найкраща з усіх можливих систем. Така стратегія проектування приводить до оптимального синтезу.

Неважко здогадатися, що оптимальний синтез неможливо виконати, якщо ми обмежимося винесенням рішення про задовільність, але не зуміємо зрівняти хоча б дві технічні системи, щоб вибрати з них кращу. Нехай запропоновано для порівняння дві технічні системи з певного набору і ми можемо зробити один з трьох висновків: перша система краща, ніж друга; друга система краща першої; з точки зору їх використання системи однакові (еквівалентні).

При кардиналістському порівнянні цих систем, оперуючи певним набором чисел і функцій, а також алгоритмом їх обробки, завжди та однозначно можна прийняти одне й тільки одне із трьох рішень. В ординалістському варіант використання набору чисел і функцій, як і алгоритмів їх обробки, не забороняється. Однак тут не вказується ніякого формального правила, за допомогою якого можна зробити однозначний Висновок про перевагу тієї чи іншої системи і який не залежав би від неформальних обставин, кваліфікації, досвіду та смаків експерта тощо.

Типовим прикладом кардиналістської оцінки може бути рішення віддати перевагу системі (приклад 4.6) з меншим S (рис. 4.3). Ординалістський шлях порівняння систем здійснюється експертом на основі вивчення графіків типу тих, які показано на рис. 4,3. Наприклад, експерт може віддати перевагу системі з характеристиками v(t), що не мають значних виступів, хоча їх відхилення від ідеальної характеристики були б значними.

4.2.4. Неоптимальний і оптимальний синтез технічних систем

Розглянемо суть неоптимального та оптимального автоматизованого синтезу технічних систем з використанням ЕОМ. Нехай опис поведінки, характеристик і властивостей системи зводиться до задання n дійсних чисел $К_1$, $К_2$, ..., $К_n$— показників якості (функціонування) технічної системи.

Кардиналістські рішення про задовільність і перевагу тих чи інших систем повністю можуть бути зведені до обробки цих п чисел, які утворюють деякий вектор К. В ординалістському трактуванні цей вектор також може виявитися корисним, хоча не повинен алгоритмічно приводити до однозначних рішень.

Неоптимальний кардиналістський синтез. Він полягає в перетворенні описів впливів, необхідної поведінки, характеристик і властивостей системи в такий опис бажаної системи, для якої одночасно виконується й нерівностей:

$$\begin{aligned} &K_{11} \le K_1 \le K_1^1; \\ &K_{21} \le K_2 \le K_2^1; \\ &\cdots\cdots\cdots\cdots \\ &K_{n1} \le K_n \le K_n^1. \end{aligned} \qquad (4.9)$$

Дійсні числа K_{i1} та K_i^1 (i = 1, 2,..., n) задаються разом зі складом вектора $\overline{K}$. Деякі з чисел K_{i1} або всі вони можуть бути і нулями. Наприклад, для механічної системи (приклад 4.7) нерівності (4.9) можуть прийняти такий вигляд:

$$\begin{aligned} &|\sigma_i| \le \sigma_i^1 \quad (i = 1, 2, \ldots, n_\sigma); \\ &|z_j| \le z_j^1 \quad (j = 1, 2, \ldots, n_z); \\ &|v_j| \le v_j^1 \quad (j = 1, 2, \ldots, n_v); \\ &|w_j| \le w_j^1 \quad (j = 1, 2, \ldots, n_w); \\ &|F_k| \le F_k^1 \quad (j = 1, 2, \ldots, n_F), \\ &\omega_1 \le \omega \text{ (нерівність з номером } n-1), \end{aligned} \qquad (4.10)$$

$$M \le M^1 \text{ (нерівність з номером } n).$$

Тут σ_i,- - напруження для і-го елемента конструкції. Абсолютна величина напруження не повинна перевищувати σ_i^1 (всього таких умов n_σ); z_j, v_j, w_j - переміщення, швидкість і прискорення в деякій j-й точці, абсолютні величини яких не повинні перевищувати граничних значень z_j^1, v_j^1, w_j^1 (таких умов n_z); F_k - зусилля в k-му елементі конструкції, яке обмежується величинами F_k^1 (число цих нерівностей n_F може дорівнювати n_σ). Передостання з нерівностей (4.10) вимагає, щоб основна частота ω не була

меншою від деякого граничного значення ω_1, а остання нерівність обмежує масу М конструкції.

Необхідно відзначити, що в нерівностях типу (4.9) часто наявні вимоги до надійності (наприклад, до напрацювання на відмову) і техніко-економічні (наприклад, обмеження зверху на вартість системи).

Найчастіше машинний синтез використовується після того, як знайдена структура системи і вибрано базис її елементів. Тоді задовольнити нерівності (4.9) - (4.10) можна, лише змінюючи вектор а, від якого залежать усі $К_1$, $К_2$,..., $К_п$.

Кардиналістський підхід до синтезу полягає в тому, що кожна система однозначно характеризується набором чисел з вектора $\overline{K}$. Рішення про задовільність системи вимагає перевірки, чи задовольняє кожний з компонентів цього вектора відповідну нерівність (4.9). Такий алгоритм синтезу не допускає ніяких неоднозначностей. Він допускає лише неоднозначність результату — може статися так, що задовільними є декілька систем. Розглянутий підхід синтезу не вказує алгоритму, за яким необхідно вибрати той чи інший варіант системи з отриманих задовільних варіантів.

Неоптимальний ординадістський синтез. Тут відсутній формальний алгоритм прийняття рішення про задовільність варіанта технічної системи. Можливий синтез системи, коли для її оцінки відсутній набір функцій і чисел. Однак такий підхід недоцільний та не використовується в практиці створення технічних систем. Більш доцільним є спосіб, коли для оцінки систем зберігається набір чисел і функцій, але процес вибору складу вектора $\overline{K}$ і крайніх меж $К_{i1}$ та K_i^1 у нерівностях (4.9) здійснюється конструктором неформально. Розглянемо це на прикладі.

Оптимальний ординалістський синтез. Співвідношення між цим підходом і тільки що розглянутим приблизно таке ж, як при неоптимальному синтезі. Як і там, перед розробником відкриваються два шляхи.

1. Можна розрахувати деякі числа, функції, побудувати якісь графіки та вважати, що вся ця інформація досить повно характеризує систему, яка

створюється. Такі розрахунки і такі побудови повторюються декілька разів у процесі зондування системи або її направленого вивчення. Потім розробник вивчає ці набори чисел і графіків для різних варіантів побудови системи. В результаті такого вивчення він робить висновок, яка з систем найкраща, оптимальна. Інший або навіть той же розробник в інших умовах може прийняти інший варіант системи — в цьому виявляється неформальність ординалістського підходу до синтезу технічних систем.

2. Можна прийти до прийняття рішення при ординалістському синтезі і більш формально, використовуючи більш вільну трактовку кардиналістських умов (4.12) та факту належності вектора а до області G_a. У процесі синтезу допускається змінювати склад нерівностей (4.9) і набір критеріїв (4.12), переводити компоненти вектора $\overline{K}$ із (4.12) у (4.9) і навпаки або знехтувати якимись з них, деформувати область G_a й змінювати межі $К_{i1}$ та K_i^1 у нерівностях (4.9).

У багатьох випадках такий ординалістський підхід виявляється більш корисним і порівняно з досить вільним ординалістськнм підходом, і порівняно з кардиналістським підходом.

4.2.5. Алгоритм неоптимального синтезу технічних систем

До цього розглядалися, в основному, проблеми постановки задачі синтезу та їх вирішення, виходячи із загальних принципів, якими можна користуватися при знаходженні описів технічних систем. Тепер розглянемо деякі конкретні способи синтезу, які базуються на жорстко алгоритмізованих крадиналістських підходах.

Нехай цілі створення нової системи й висунуті до неї вимоги добре відомі розробнику, і він може визначити показники якості $К_I$, $К_2$, ..., $К_n$, які повністю описують систему, і можна вказати межі $К_{11}$, K_1^1, ..., $К_{n1}$, K_n^1, в яких відповідні показники повинні знаходитись. Кардиналістська постановка

задачі породжує абсолютно жорсткі вимоги до майбутньої системи. В обмін на це розробник отримує можливість перекласти весь процес синтезу технічної системи на ЕОМ, оскільки ніяких неформальних рішень, що пов'язані зі зміною постановки задачі, йому приймати не доведеться, хіба що в такій постановці розв'язати проблему синтезу взагалі не вдається.

Процес синтезу розпадається на ряд стадій. Перші з них пов'язані з синтезом структури і вибором базису елементів, тобто набору елементів, із яких будуються системи. Ці стадії погано піддаються формалізації, і, як правило, виконуються людиною при допомозі ЕОМ. Однак покажемо, як міг би бути алгоритмізований синтез структури, або, як його ще називають, структурний синтез.

Нехай перетворення деякого входу u(t) у вихід y(t) у довільній за фізичною природою системі здійснюється всього лише трьома блоками A, B, C і кожен з них розміщується за іншим так, як показано на рис. 4.5 (приклад 4.8). Кожний із блоків A, B, C може займати будь-яку позицію 1, 2, 3.

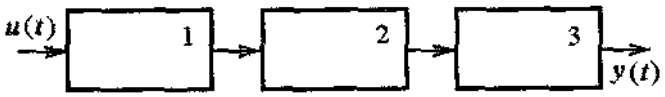

Рис. 4.5. Структура системи, що синтезується: 1, 2, 3 — номери блоків системи

У цій дуже ідеалізованій ситуації на кожен із блоків можна дивитися як на неподільний елемент системи. В сукупності блоки A, B, C складають базис елементів синтезу. При цьому ми вважаємо, що мікроструктура кожного блока вибрана і фіксована, але їх параметри не визначені та є вільними. Позначимо відповідні сукупності параметрів блоків через a_A, a_B, a_c.

До чого в таких умовах зводиться синтез системи? По-перше, до вибору порядку розміщення блоків A, B, C: можливі структури ABC, BCA, CAB, CBA, BAC, ACB. По-друге, до знаходження всіх компонентів вектора й, тобто до знаходження "підвекторів" a_A, a_B, a_c. Результат останнього

розрахунку повинен залежати від порядку розміщення блоків. Тому визначення параметрів, яке називається параметричним синтезом, не може бути ізольовано від синтезу структури.

До початкової інформації в кардиналістському неформальному синтезі належать: опис структур елементів (у нашому прикладі блоків A, B, C); опис області G_a, до якої можуть належати допустимі значення набору параметрів a , тобто об'єднання наборів a_A, a_B, a_c; 2n чисел $К_{11}$, K_1^1, ..., $К_{n1}$, K_n^1, які входять у нерівність (4.9) і визначають уже сформульовані умова прийнятності системи, що синтезується.

Відомо, що синтез становить перетворення одних описів в інші. Описи, які нам треба знайти, такі: трьохелементна послідовність символів, що розміщує в певному порядку букви A, B, C — це опис структури системи; набір чисел $\overline{a}$, тобто опис параметрів системи. Задача синтезу буде розв'язана, якщо описи, що знаходяться в початковій інформації, виявляться перетворєними в останні два.

Алгоритм синтезу можна розділити на наступні етапи.

1. На основі інформації про подібні системи й спираючись на досвід та інтуїцію конструктора або випадково, останній видає певну послідовність символів A, B, C і деякий початковий вектор параметрів $\overline{a}^{01}$. Цим повністю визначається система, якщо дотримуватися початкових припущень.

2. Проводиться частковий аналіз системи, що вибрана на попередньому етапі. Такий аналіз обмежується розрахунком вектора $\overline{K}$.

3. Здійснюється перевірка n нерівностей (4.9) і формується висновок про задовільність чи незадовільність системи.

Якщо система виявилась задовільною, то процес неоптимального синтезу можна вважати вдало завершеним. Якщо нерівності (4.9) не виконані, то здійснюється новий процес підбору задовільної системи. Цей процес здійснюється шляхом повернення до першого етапу при наявності інформації про попередню незадовільну систему.

Можливі варіанти алгоритму синтезу відрізняються стратегією вибору в

цих умовах нової системи. Тут можна йти двома напрямами. Перший — зберегти попередню структуру системи, тобто послідовність блоків, і постаратися задовольнити умови придатності системи за рахунок більш вдалого вибору вектора параметрів a, ніж у попередньому випадку, коли інформації було менше, ніж зараз. Це, по суті, намагання вийти з положення за рахунок параметричного синтезу. Другий — вважати, що незадовільність системи залежить від поганої структури, і тому подальші пошуки в просторі параметрів дають мало шансів на успіх. Тоді структура змінюється, тобто вибирається нова послідовність елементів A, B, C і в цій структурі вибирається нова комбінація параметрів — вектор $\overline{a}$.

Якщо в нових умовах на першому етапі нова система так або інакше вибрана, то можна перейти до етапу 3 і проаналізувати на задовільність отриману систему. При цьому можливі два варіанти: або запропонований алгоритм синтезу приведе до задовільної системи, або обчислювальні ресурси будуть вичерпані до того, як це станеться (в подібних циклічних розрахунках часто встановлюється певний граничний час, і ЕОМ автоматично закінчує розрахунок, як тільки відведений час використано). В другому випадку розробнику необхідно замінити початкову інформацію, розширити обмеження на структуру або перейти до нового алгоритму синтезу.

Подібні розв'язки вимагають високої кваліфікації розробника. Перехід до нового алгоритму базується на впевненості, цю задача синтезу в її початковій кардиналістській постановці може бути розв'язана, але використаний раніше алгоритм або не знаходить розв'язку, або розшукує його недопустимо повільно. Можливий також варіант, що не існує жодної структури й жодної комбінації параметрів, які дають змогу задовольнити й нерівностей (4.9). Тоді всі подальші намагання приведуть до невдачі і обчислювальні ресурси будуть використані марно.

4.2.6. Правила зміни структури і параметрів технічних систем

Правила зміни структури можуть базуватися на двох різних ідеях. Перша полягає в довільному переборі всіх можливих структур. Це, по суті, зондування структурного простору. Друга ідея враховує інформацію, що накопичилась на останньому або декількох попередніх спробах синтезу. Нехай у першій серії спроб останнього прикладу 4.8 була структура АВС і намагання знайти для неї такі параметри, щоб система стала задовільною, виявились безуспішними. Алгоритм подальшого пошуку структури може будуватися з урахуванням невдачі в першій спробі. З неї, наприклад, може випливати, що А не повинно бути першим блоком. Тоді структуру АСВ можна не розглядати і таким чином зменшити кількість спроб синтезу структури системи.

Такі алгоритми можуть базуватися лише на глибокому розумінні того, як функціонує система, як структура першого блоку впливає на вектор показників $\overline{K}$ усієї системи і т. д. Зрозуміло також, що невдача першої спроби може бути пов'язана з двома наступними блоками і подальший синтез системи в просторі структур буде направлено по хибному шляху.

Необхідно відзначити, що структурний синтез вимагає значних зусиль та витрат, бо його перспективність стає зрозумілою тільки тоді, коли закінчено і параметричний синтез. Оскільки структури містять, як правило, досить велику кількість елементів, то вимоги до обчислювальних засобів бувають досить значними.

Алгоритмічні пошуки структури системи пов'язані зі значними труднощами, їх можна порівняти, наприклад, з пошуком найкращого наступного ходу в шаховій партії. Для них характерні і вимушена відмова від довільного перебору всіх варіантів, і необхідність кількісно оцінити значну й важко формалізовану накопичену інформацію, і проблема прогнозу тих наслідків, до яких може привести прийняте рішення. Такі задачі зараз інтенсивно досліджуються, і, можливо, з часом перспективи алгоритмічних

пошуків структури будуть більш реалістичними, ніж сьогодні.

Ситуація з правилами зміни параметрів при синтезі технічних систем значно спрощується. В цій задачі структура системи відома, залишається знайти такий вектор а, щоб були виконані n нерівностей (4.9). Оскільки в усіх відношеннях, крім вибору вектора, система описана, то відомі і правила обчислення вектора $\overline{K}$.

Отож, формально кажучи, необхідно розв'язати деяку систему нерівностей. Розв'язуванням цієї задачі займається спеціальна галузь математики. Покажемо, як поєднати розв'язування системи нерівностей із задачею знаходження екстремуму деякої спеціально підібраної функції. Таке зведення однієї проблеми до іншої поєднує між собою неоптимальний і оптимальний параметричний синтез систем. Пояснюється це тим, що багато задач оптимального синтезу також зводяться до знаходження екстремуму.

4.3. Морфологічний аналіз і синтез технічних систем

Метод морфологічного аналізу та синтезу, розроблений швейцарським астрономом Ф.Цвіккі, побудований на принципах комбінаторики [27]. Суть його полягає в тому, що в технічній системі або в іншому об'єкті виділяють групу основних конструктивних або інших ознак. Для кожної ознаки вибирають альтернативні варіанти, тобто можливі варіанти його реалізації. Комбінуючи їх між собою, можна отримати множину різних технічних рішень, у тому числі І тих рішень, які мають практичний інтерес.

Практичне використання методу полягає в побудові морфологічної таблиці, заповненні її можливими альтернативними варіантами та виборі із всієї множини найбільш прийнятних технічних рішень.

Найбільше поширення отримав метод Цвіккі, в якому за ознаки вибираються функції елементів технічної системи, а за альтернативні варіанти — різні способи реалізації кожної функції. В цьому випадку морфологічна таблиця буде мати стільки стовпців, скільки функціональних

елементів в системі на вибраному рівні. Тоді кількість можливих варіантів технічних рішень визначається залежністю

$$N = n_1 \cdot n_2 \cdot \ldots \cdot n_m,$$

де m – число функціональних елементів системи на заданому рівні; n_i – число альтернативних варіантів і-го (i = 1, 2, ..., m) функціонального елемента.

Розглянемо побудову морфологічної таблиці на прикладі стрілової системи вантажопідйомного крана з горизонтальним переміщенням вантажу при зміні вильоту (рис. 4.7).

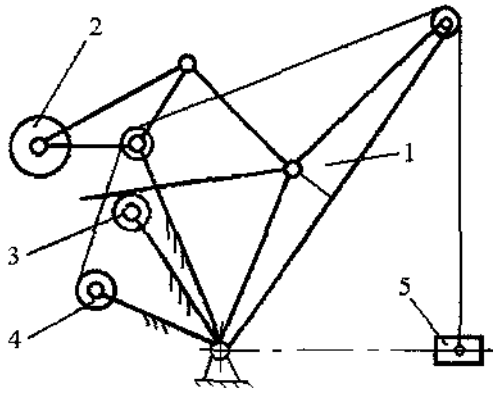

Рис. 4.7. Схема стрілової системи крана

Стрілова система складається з таких основних функціональних елементів: 1 - стріловий пристрій; 2 - механізм його врівноваження; 3 - приводний механізм; 4 - механізм вирівнювання траєкторії вантажу 5. Для розглянутої системи морфологічна таблиця має чотири стовпці, в які входять функціональні елементи стрілової системи (табл. 4.1). У цій же таблиці в кожному стовпці приведені альтернативні варіанти функціональних елементів стрілової системи.

Шляхом вибору одного з альтернативних варіантів технічних рішень з кожного стовпця отримаємо один із можливих варіантів стрілової системи, Так, якщо взяти альтернативні варіанти під першим номером із кожного стовпця, то отримаємо варіант стрілової системи, який показано на рис. 4.7.

Усього ж з цієї таблиці можна отримати N = 6·5·6·5=900 варіантів системи вантажопідйомного крана з горизонтальним переміщенням вантажу. Ця таблиця може доповнюватися новими можливими варіантами функціональних елементів.

Таблиця 4.1 Морфологічна таблиця можливих варіантів стрілової системи вантажопідйомного крана

Альтер-нативні варіанти	Функціональні елементи			
	Стріловий пристрій	Механізм врівноваження	Приводний механізм	Механізм вирівнювання траєкторії вантажу
1	Жорстка прямолінійна стріла	Чотири-ланковий	Рейковий	Профільний барабан
2	Жорстка криволінійна стріла	Шести-ланковий	Гвинтовий	Вирівнюваль-ний блок
3	Шарнірно-складова стріла з прямолінійним хоботом	Поліспас-товий	Гідравлічний	Вирівнюваль-ний поліспаст
4	Шарнірно-складова стріла з профільним хоботом	На стрілі	Кривошипний	Вантажний канат, пара-лельний осі стріли
5	Прямолінійна стріла з висувними секціями	Урівнова-жувальни й візок	Поліспастовий	Вантажний канат, паралельний осі відтяжки
6	Прямолінійна стріла із вставними секціями	—	Кривошипно-коромисловий	—

Розглянемо процес складання морфологічної таблиці на більш низькому рівні для приводного механізму стрілової системи вантажопідйомного крана. Одним із варіантів приводного механізму може бути рейковий механізм (рис. 4.8).

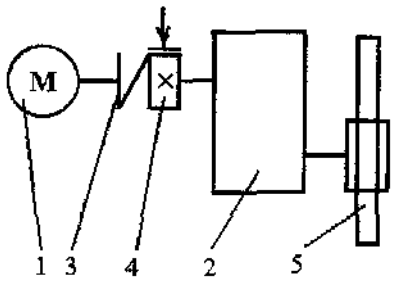

Рис. 4.8. Кінематична схема рейкового приводного механізму

Цей приводний механізм складається з двигуна - 1, передавального механізму - 2, з'єднувального пристрою - 3, гальмівного механізму - 4 та рейкового механізму -5.

Таблиця 4.2 Морфологічна таблиця можливих варіантів приводного механізму стрілової системи вантажопідйомного крана

Альтер-нативні варіанти	Функціональні елементи				
	Двигун	Передавальний механізм	З'єднувальний пристрій	Гальмівний механізм	Виконавчий механізм
1	Електродвигун постійного струму	Циліндричний редуктор	Втулочно-пальцева муфта	Колодковий з електро-магнітним штовхачем	Зубчастий
2	Електродвигун змінного струму з коротко-замкнутим ротором	Планетарний редуктор	Зубчаста муфта	Колодковий з гідравлічним штовхачем	Цівковий
3	Електродвигун змінного струму з контактними кільцями	Черв'ячний редуктор	Ланцюгова муфта	Стрічковий з важельним простим керуванням	Черв'ячний

Альтер-нативні варіанти	Функціональні елементи				
	Двигун	Передавальний механізм	З'єднувальний пристрій	Гальмівний механізм	Виконавчий механізм
4	Гідродвигун	Хвильовий редуктор	Кулачково-дискова муфта	Стрічковий диференціальний	Гвинтовий
5	Двигун внурішньо-го згоряння	Конічний редуктор	Гідромуфта	Дисковий	Поліспастовий
6	—	Комбінований редуктор	Глухе фланцеве з єднання	Порошко-вий електро-магнітний	Рейковий
7	—	—	Кулачковий	Електро-індукційний	Кривошип-но-коромисло-вий

У приводному механізмі видалено п'ять функціональних елементів, різні альтернативні варіанти яких утворюють морфологічну таблицю (табл. 4.2).

Аналіз табл. 4.2 показує, що можна отримати N = 5·6·7·7·7 = 10290 варіантів приводу зміни вильоту стрілової системи. Не всі ці варіанти приводу можуть мати практичне втілення Однак значна кількість альтернативних варіантів дає змогу провести аналіз різних конструктивних рішень І вибирати з них у тих чи інших умовах найбільш ефективні та перспективні конструкції приводів.

5. РОЗВИТОК МЕТОДОЛОГІЇ ПРОГРАМУВАННЯ, ПРОЕКТУВАННЯ ТА МОДЕЛЮВАННЯ ТЕХНІЧНИХ СИСТЕМ

Якщо спробувати охарактеризувати сучасний рівень розвитку комп'ютерних та інформаційних технологій, то перше, на що слід звернути увагу - це зростаюча складність не тільки окремих фізичних і програмних компонентів, але і лежать в основі цих технологій концепцій та ідей. Здається, ще зовсім недавно професійному програмістові було досить досконало володіти одним-двома мовами програмування, щоб розробляти серйозні програмні додатки. Вибір платформи і операційної системи, як правило, не був серйозною проблемою. А супровід програми, хоча і було пов'язане з об'єктивними труднощами, могло бути реалізовано простим додаванням або зміною коду вихідної програми.

5.1. Процедурно-орієнтоване програмування

Поява перших електронних обчислювальних машин або комп'ютерів ознаменувало новий етап у розвитку техніки обчислень. Здавалося, досить розробити послідовність елементарних дій, кожне з яких перетворити в зрозумілі комп'ютеру інструкції, і будь-яка обчислювальна задача може бути вирішена. Ця ідея виявилася настільки життєздатною, що довгий час домінувала над усім процесом розробки програм. З'явилися спеціальні мови програмування, які дозволили перетворювати окремі обчислювальні операції в відповідний програмний код.

Основою даної методології розробки програм була процедурна або алгоритмічна організація структури програмного коду. Це було настільки природно для вирішення обчислювальних задач, що ні у кого не викликала сумнівів доцільність такого підходу. Вихідним поняттям цієї методології було поняття алгоритму, під яким, в загальному випадку, розуміється деякий припис виконати точно певну послідовність дій, спрямованих на досягнення

заданої мети або рішення поставленого завдання. Прикладами алгоритмів є добре відомі правила знаходження коренів квадратного рівняння або коренів лінійної системи рівнянь.

З цієї точки зору вся історія математики тісно пов'язана з розробкою тих чи інших алгоритмів вирішення актуальних для своєї епохи завдань. Більш того, саме поняття алгоритму стало предметом відповідної теорії - теорії алгоритмів, яка займається вивченням загальних властивостей алгоритмів. Згодом зміст цієї теорії стало настільки абстрактним, що відповідні результати розуміли тільки фахівці. Як данина цієї традиції якийсь період часу мови програмування називалися алгоритмічними, а перше графічне засіб документування програм отримало назву блок-схеми алгоритму. Відповідна система графічних позначень була зафіксована в ГОСТ 19.701-90, який регламентував використання умовних позначень в схемах алгоритмів, програм, даних і систем.

Однак потреби практики не завжди вимагали встановлення обчислюваності конкретних функцій або розв'язання окремих завдань. У мовах програмування виникло і закріпилося нове поняття процедури, яке конкретизував загальне поняття алгоритму стосовно до вирішення завдань на комп'ютерах. Так само, як і алгоритм, процедура являє собою закінчену послідовність дій або операцій, спрямованих на рішення окремого завдання. У мовах програмування з'явилася спеціальна синтаксична конструкція, яка отримала назву процедури.

Згодом розробка великих програм перетворилася в серйозну проблему і поставила вимогу про їх розбиття на більш дрібні фрагменти. Основою для такого розбиття якраз і стала процедурна декомпозиція, при якій окремі частини програми або модулі представляли собою сукупність процедур для вирішення деякої сукупності завдань. Головна особливість процедурного програмування полягає в тому, що програма 'завжди має початок в часі або початкову процедуру (початковий блок) і закінчення (кінцевий блок). При

цьому вся програма може бути представлена візуально у вигляді спрямованої послідовності графічних примітивів або блоків (рис. 1.1).

Важливою властивістю таких програм є необхідність завершення всіх дій попередньої процедури для початку дій подальшої процедури. Зміна порядку виконання цих дій навіть в межах однієї процедури зажадало включення в мови програмування спеціальних умовних операторів типу if-then-else і Goto для реалізації розгалуження обчислювального процесу в залежності від проміжних результатів рішення задачі.

Рис. 5.1. Графічне представлення програми у вигляді послідовності процедур

Розглянуті ідеї сприяли становленню певної системи поглядів на процес розробки програм і написання програмних кодів, яка Отримала назву методології структурного програмування. Основою даної методології є процедурна декомпозиція програмної системи і організація окремих модулів у вигляді сукупності виконуваних процедур. В рамках даної методології отримало розвиток спадний проектування програм або програмування "зверху-вниз". Період найбільшої популярності ідей структурного програмування доводиться на кінець 70-х-початок 80-х років.

Як допоміжний засіб структуризації програмного коду було рекомендовано використання відступів на початку кожного рядка, які повинні

виділяти вкладені цикли і умовні оператори. Все це покликане сприяти розумінню або читабельності самої програми. Дане правило згодом було реалізовано в сучасних інструментаріях розробки програм.

У цей період основним показником складності розробки програм вважали її розмір. Цілком серйозно обговорювалися такі оцінки складності програм, як кількість рядків програмного коду. Правда, при цьому робилися деякі припущення щодо синтаксису самих рядків, які повинні були відповідати певним правилам. Загальна трудомісткість розробки програм оцінювалася спеціальної одиницею виміру - "людино-місяць" або "людино-рік". А професіоналізм програміста безпосередньо зв'язувався з кількістю рядків програмного коду, який він міг написати й налагодити протягом, скажімо, місяця.

5.2. Об'єктно-орієнтоване програмування

Згодом ситуація стала істотно змінюватися. Виявилося, що трудомісткість розробки програмних додатків на початкових етапах програмування оцінювалася значно нижче реально витрачених зусиль, що служило причиною додаткових витрат і затягування остаточних термінів готовності програм. В процесі розробки додатків змінювалися функціональні вимоги замовника, що ще більш віддаляло момент закінчення роботи програмістів. Збільшення розмірів програм призводило до необхідності залучення більшої кількості програмістів, що, в свою чергу, потребувало додаткових ресурсів для організації їх узгодженої роботи.

Але не менш важливими виявилися якісні зміни, пов'язані зі зміщенням акценту використання комп'ютерів. Якщо в епоху "великих машин" основними споживачами програмного забезпечення були великі підприємства, компанії і установи, то пізніше з'явилися персональні комп'ютери і стали повсюдним атрибутом дрібного і середнього бізнесу. Обчислювальні та розрахунково-алгоритмічні завдання в цій галузі

традиційно займали другорядне місце, а на перший план виступили завдання обробки і маніпулювання даними.

Стало очевидним, що традиційні методи процедурного програмування не здатні впоратися ні зі зростаючою складністю програм і їх розробки, ні з необхідністю підвищення їх надійності. У другій половині 80-х років виникла нагальна потреба в новій методології програмування, яка була б здатна вирішити весь цей комплекс проблем. Такий методологією стало об'єктно-орієнтоване програмування (ООП).

Фундаментальними поняттями ООП є поняття класу і об'єкту. При цьому під класом розуміють деяку абстракцію сукупності об'єктів, які мають загальний набір властивостей і мають однаковий поведінкою. Кожен об'єкт в цьому випадку розглядається як екземпляр відповідного класу. Об'єкти, які не мають абсолютно однакових властивостей або не володіють однаковим поведінкою, за визначенням, не можуть бути віднесені до одного класу.

Важливою особливістю класів є можливість їх організації у вигляді деякої ієрархічної структури, яка за зовнішнім виглядом нагадує схему класифікації понять формальної логіки. У зв'язку з цим слід зауважити, що кожне поняття в логіці має певний обсяг і зміст. При цьому під обсягом поняття розуміють всі інші мислимі поняття, для яких вихідне поняття може служити визначальною категорією або головною частиною. Зміст поняття становить сукупність усіх його ознак або атрибутів, що відрізняють дане поняття від всіх інших. У формальній логіці має місце закон зворотного відносини: якщо зміст поняття А міститься в змісті поняття В, то обсяг поняття В міститься в обсязі поняття А.

Ієрархія понять будується наступним чином. В якості найбільш загального поняття або категорії береться поняття, що має найбільший обсяг і, відповідно, найменше зміст. Це найвищий рівень абстракції для даної ієрархії. Потім дане загальне поняття деяким чином конкретизується, тим самим зменшується його обсяг і збільшується вміст. З'являється менш загальне поняття, яке на схемі ієрархії буде розташовано на рівень нижче

вихідного поняття. Цей процес конкретизації понять може бути продовжений до тих пір, поки на самому нижньому рівні не буде отримано поняття, подальша конкретизація якого в даному контексті або неможлива, або недоцільна.

Прикладами найбільш загальних понять можуть служити такі абстрактні категорії, як система, структура, інтелект, інформація, сутність, зв'язок, стан, подія і багато інших. У процесі вивчення цих категорій з'являються нові особливості їх змісту та обсягу. Саме з цих причин завжди важко дати їм точне визначення. Як приклади конкретних понять можна привести поняття книги, яку читач тримає в руках, або поняття мікропроцесора Intel Pentium П-300.

Основними принципами ООП є спадкоємство, інкапсуляція і поліморфізм. Принцип, відповідно до якого знання про більш загальної категорії дозволяється застосовувати для більш вузької категорії, називається спадкуванням. Спадкування тісно пов'язане з ієрархією класів, яка визначає, які класи слід вважати найбільш абстрактними і загальними по відношенню до інших класів. При цьому, якщо деякий більш загальний або батьківський клас (предок) володіє фіксованим набором властивостей і поведінкою, то похідний від нього клас (нащадок) повинен містити цей же набір властивостей і поведінку, а також додаткові, які будуть характеризувати унікальність отриманого таким чином класу. У цьому випадку говорять, що похідний клас успадковує властивості і поведінку батьківського класу.

Для ілюстрації принципу успадкування можна навести такий приклад. Розглянемо в якості загального клас "Автомобіль". Даний клас визначається як деяка абстракція властивостей і поведінки всіх реально існуючих автомобілів. При цьому властивостями класу "Автомобіль" можуть бути такі загальні властивості, як наявність двигуна, трансмісії, коліс, рульового управління. Якщо в якості похідного класу розглянути клас "Легковий автомобіль", то всі виділені вище властивості будуть властиві й цього класу. Можна сказати, що клас "Легковий автомобіль" успадковує властивості

батьківського класу "Автомобіль". Однак, крім перерахованих властивостей, клас-нащадок буде містити додаткові властивості, наприклад таке, як наявність салону з кількістю посадочних місць 2-5.

У свою чергу, клас "Легковий автомобіль" здатний породжувати інші підкласи, які цілком можуть відповідати, наприклад, моделям конкретних фірм-виробників. Таким чином, можна розглядати клас "Легковий автомобіль виробництва ВАЗ". Оскільки Волзький автомобільний завод випускає кілька моделей автомобілів, одним з похідних класів для попереднього класу може бути конкретна модель автомобіля, наприклад, ВАЗ-21083. Нарешті, виготовлений автомобіль має унікальний заводський номер, який відрізняє один автомобіль від іншого. Таким номером може бути, наприклад, ХТА-210830S1594301. В останньому випадку клас буде складатися з єдиного об'єкта або примірника, яким буде легковий автомобіль виробництва ВАЗ із зазначеним вище заводським номером.

Описана вище інформація про співвідношення класів в нашому прикладі володіє одним серйозним недоліком, а саме відсутністю наочності. У зв'язку з цим виникає питання: а чи можливо уявити ієрархію спадкування класів у візуальній формі? Традиційно для зображення понять у формальній логіці використовувалися окружності або прямокутники. Тоді для розглянутого прикладу ієрархія породження класів може бути представлена у вигляді вкладених прямокутників, кожен з яких відповідає окремому класу (рис. 5.2).

Поява об'єктно-орієнтованих мов програмування було пов'язано з необхідністю реалізації концепції класів і об'єктів на синтаксичному рівні. З точки зору ООП клас є подальшим розширенням структури (structure) або записи (record). Включення в відомі мови програмування C і Pascal класів і деяких інших можливостей призвело до появи відповідно C ++ і Object Pascal, які на сьогодні є найбільш поширеними мовами розробки додатків.

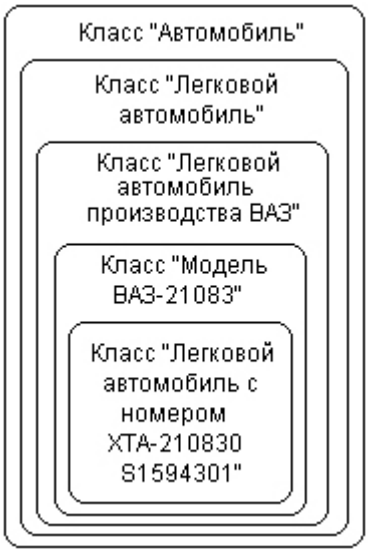

Рис. 5.2. Ієрархія вкладеності класів для прикладу "Автомобіль"

Поширенню C ++ і Object Pascal сприяла та обставина, що мова C ++ був обраний в якості базового для програмного інструментарію MS Visual C ++, а мова Object Pascal- для популярного засобу швидкої розробки додатків Borland / Inprise Delphi.

За короткий період часу обидва інструментарію перетворилися в потужні системи розробки програм з відповідними бібліотеками стандартних класів, що містять сотні різних властивостей і методів. Стосовно до середовища MS Visual C ++ 5/6 така бібліотека має спеціальну назву - MFC (Microsoft Foundation Classes), т. Е. Фундаментальні класи від Microsoft. При цьому похідні класи успадковують властивості і методи батьківських класів. Нижче наводиться фрагмент ієрархії класів MFC в тому вигляді, як він зображений у відповідній документації (рис. 5.3).

Процес розробки програм в середовищі Borland / Inprise Delphi також тісно пов'язаний з використанням бібліотеки стандартних класів - VCL (Visual Component Library) або бібліотеки візуальних компонентів. Ця бібліотека теж побудована за ієрархічним принципом, відповідно до якого компоненти нижчих рівнів успадковують властивості і методи верхніх компонентів. Для даного випадку також наводиться фрагмент ієрархії класів VCL (рис. 5.4).

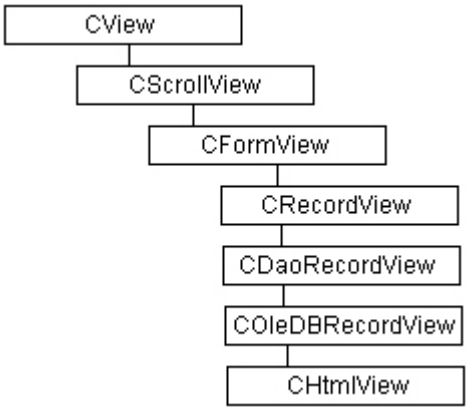

Рис. 5.3. Фрагмент ієрархії класів MFC, використовуваних в середовищі програмування MS Visual C ++ 5/6

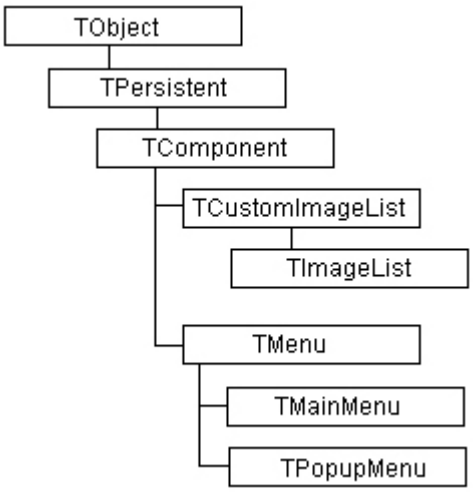

Рис. 5.4. Фрагмент ієрархії класів VCU використовуваних в середовищі програмування Borland / Inprise Delphi 3-4

Навіть цих простих прикладів досить, щоб зрозуміти наступний факт. А саме, для однієї і тієї ж загальної концепції ієрархії класів використовуються абсолютно різні графічні засоби. У першому випадку - вкладені прямокутники, у другому - зв'язкові прямокутники. Насправді різних способів зображення класів запропоновано набагато більше, невелика частина з них буде розглянута нижче. Однак уже зараз важливо усвідомити, що подібну ситуацію слід було б уніфікувати, т. Е. Використовувати для цієї мети деяку єдину систему позначень.

Наступний принцип ООП - інкапсуляція. Цей термін характеризує приховування окремих деталей внутрішнього устрою класів від зовнішніх по відношенню до нього об'єктів або користувачів. Дійсно, взаимодействующему з класом суб'єкту або клієнту немає необхідності знати, яким чином реалізований той чи інший метод класу, послугами якого він вирішив скористатися. Конкретна реалізація властивих класу властивостей і методів, які визначають поведінку цього класу, є власною справою даного класу. Більш того, окремі властивості і методи класу взагалі можуть бути невидимі за межами цього класу, що є базовою ідеєю введення різних категорій видимості для компонентів класу.

Якщо продовжити розгляд прикладу з класом "Легковий автомобіль", то неважко проілюструвати інкапсуляцію наступним чином. Основним суб'єктом, який взаємодіє з цим класом, є водій. Цілком очевидно, що не кожен водій досконало знає внутрішній устрій легкового автомобіля. Більш того, окремі деталі цього пристрою свідомо приховані в корпусі двигуна або в коробці передач. А в разі порушення роботи автомобіля, що є причиною неадекватності його поведінки, необхідний ремонт виконує професійний механік.

Інкапсуляція веде своє походження від ділення модулів в деяких мовах програмування на дві частини або секції: інтерфейс і реалізацію. При цьому в інтерфейсний секції модуля описуються всі оголошення функцій і процедур, а можливо і типів даних, доступних за межами даного модуля. Іншими словами, зазначені процедури і функції є способами надання послуг зовнішнім клієнтам. В іншій секції модуля, званої реалізацією, міститься програмний код, який визначає конкретні способи реалізацій оголошених в інтерфейсний частини процедур і функцій.

Принцип поділу модуля на інтерфейс і реалізацію відображає суть наших уявлень про навколишній світ. В інтерфейсній частині вказується вся інформація, необхідна для взаємодії з будь-якими іншими об'єктами.

Реалізація приховує або маскує від інших об'єктів всі деталі, що не мають відношення до процесу взаємодії об'єктів (рис. 5.5).

Рис. 5.5. Ілюстрація приховування внутрішніх деталей реалізації методів класів

Третім принципом ООП є поліморфізм. Під поліморфізмом (грец. Poly- багато, morfos - форма) розуміють властивість деяких об'єктів приймати різні зовнішні форми в залежності від обставин. Стосовно до ООП поліморфізм означає, що дії, які виконуються однойменними методами, можуть відрізнятися в залежності від того, якого з класів належить той чи інший метод.

Розглянемо, наприклад, три об'єкта або класу: двигун автомобіля, електричне світло в кімнаті і персональний комп'ютер. Для кожного з них можна визначити операцію "вимкнути". Однак сутність цієї операції буде відрізнятися для кожного з розглянутих об'єктів. Так для двигуна автомобіля виклик методу двігатеяь_автомобіля. вимкнути про означає припинення подачі палива і його зупинку. Виклик методу Кімната. електріческій_ світло. вимкнути про означає простий клацання вимикача, після чого кімната зануриться в темряву. В останньому випадку дію персональний_ комп'ютер. вимкнути про може бути причиною втрати даних, якщо виконується нерегламентованим чином.

У нашому прикладі для операції вимкнути () можна визначити такі додаткові параметри, як час вимикання, деякий умова знаходження об'єкта в попередньо включеному стані і ін. Для цього після імені операції вказуються дужки, в яких можуть бути вказані ці додаткові параметри або аргументи. У

разі відсутності аргументів вважається, що список параметрів порожній. Однак дужки все одно записуються і вказують на той факт, що відповідне ім'я є ім'ям операції або методу, на відміну від властивостей або атрибутів класу, які записуються без дужок.

Поліморфізм об'єктно-орієнтованих мов пов'язаний з перевантаженням функцій, але не тотожний їй. Важливо мати на увазі, що імена методів і властивостей тісно пов'язані з класами, в яких вони описані. Ця обставина забезпечує певну надійність роботи програми, оскільки виключає випадкове застосування методу для вирішення невластивою йому завдання.

Широке поширення методології ООП вплинуло на процес розробки програм. Зокрема, процедурно-орієнтована декомпозиція програм поступилася місцем об'єктно-орієнтованої декомпозиції, при якій окремими структурними одиницями програми стали є не процедури і функції, а класи та об'єкти з відповідними властивостями і методами. Як наслідок, програма перестала бути послідовністю визначених на етапі кодування дій, а стала подієво-керованої. Остання обставина стала домінуючим при розробці широкого кола сучасних додатків. У цьому випадку кожна програма являє собою нескінченний цикл очікування деяких заздалегідь визначених подій. Ініціаторами подій можуть бути інші програми або користувачі. При настанні окремого події, наприклад, натискання клавіші на клавіатурі або клацання кнопкою миші, програма виходить зі стану очікування і реагує на всі ці події цілком адекватним чином. Реакція програми при цьому теж зв'язується з подальшими подіями.

Найбільш істотним обставиною у розвитку методології ООП стало усвідомлення того факту, що процес написання програмного коду може бути відділений від процесу проектування структури програми. Дійсно, до того як почати програмування класів, їх властивостей і методів, необхідно визначити, чим же є самі ці класи. Більш того, потрібно дати відповіді на такі питання, як: скільки і які класи потрібно визначити для вирішення поставленого

завдання, які властивості і методи необхідні для додання класів необхідного поводження, а також встановити взаємозв'язок між класами.

Ця сукупність завдань не стільки пов'язана з написанням коду, скільки з загальним аналізом вимог до майбутньої програми, а також з аналізом конкретної предметної області, для якої розробляється програма. Всі ці обставини призвели до появи спеціальної методології, що отримала назву методології об'єктно-орієнтированнного аналізу і проектування (ООАП).

5.3. Методологія об'єктно-орієнтованого аналізу і проектування

Необхідність аналізу предметної області до початку написання програми була усвідомлена давно при розробці масштабних проектів. Процес розробки баз даних істотно відрізняється від написання програмного коду для вирішення обчислювальної задачі. Головна відмінність полягає в тому, що при проектуванні бази даних виникає необхідність в попередній розробці концептуальної схеми, яка відображала б загальні взаємозв'язки предметної області та особливості організації відповідної інформації. При цьому під предметною областю прийнято розуміти ту частину реального світу, яка має суттєве значення або безпосереднє відношення до процесу функціонування програми. Іншими словами, предметна область включає в себе тільки ті об'єкти і взаємозв'язки між ними, які необхідні для опису вимог і умов вирішення деякої задачі.

Виділення вихідних або базових компонентів предметної області, необхідних для вирішення того чи іншого завдання, представляє, в загальному випадку, нетривіальну проблему. Складність даної проблеми проявляється в неформальному характері процедур або правил, які можна застосовувати для цієї мети. Більш того, така робота повинна виконуватися спільно з фахівцями або експертами, які добре знають предметну область. Наприклад, якщо розробляється база даних для обслуговування пасажирів великого аеропорту, то в проектуванні концептуальної схеми бази даних

повинні брати участь штатні співробітники даного аеропорту. Ці співробітники повинні добре знати весь процес обслуговування пасажирів або дану предметну область.

Для виділення або ідентифікації компонентів предметної області було запропоновано кілька способів і правил. Сам цей процес отримав назву концептуалізації предметної області. При цьому під компонентою розуміють деяку абстрактну одиницю, яка володіє функціональністю, т. Е. Може виконувати певні дії, пов'язані з вирішенням поставлених завдань. На попередньому етапі концептуалізації рекомендується використовувати так звані CRC-картки (Component, Responsibility, Collaborator- компонента, обов'язок, співробітники) [1]. Для кожної виділеної компоненти предметної області розробляється власна CRC-картка (рис. 5.6).

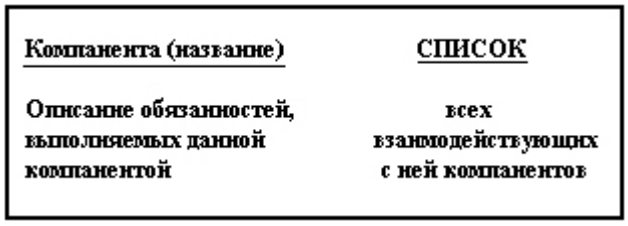

Рис. 5.6. Загальний вигляд CRC-картки для опису компонентів предметної області

Поява методології ООАП зажадало, з одного боку, розробки різних засобів концептуалізації предметної області, а з іншого - відповідних фахівців, які володіли б цією методологією. На даному етапі з'являється відносно новий тип фахівця, який отримав назву аналітика або архітектора. Поряд з фахівцями з предметної області аналітик бере участь в побудові концептуальної схеми майбутньої програми, яка потім перетворюється програмістами в код. При цьому окремі компоненти вибираються таким чином, щоб при подальшій розробці їх було зручно представити у формі класів і об'єктів. У цьому випадку важливе значення набуває і сама мова представлення інформації про концептуальну схему предметної області.

Поділ процесу розробки складних програмних додатків на окремі етапи сприяло становленню концепції життєвого циклу програми. Під життєвим циклом (ЖЦ) програми розуміють сукупність взаємопов'язаних і наступних у часі етапів, починаючи від розробки вимог до неї і закінчуючи повною відмовою від її використання. Стандарт ISO / IEC 12207, хоча і описує загальну структуру ЖЦ програми, не конкретизує деталі виконання тих чи інших етапів. Згідно з прийнятими поглядами ЖЦ програми складається з наступних етапів:

- Аналізу предметної області і формулювання вимог до програми
- Проектування структури програми
- Реалізації програми в кодах (власне програмування)
- впровадження програми
- супроводу програми
- Відмови від використання програми

На етапі аналізу предметної області та .формулiровкі вимог здійснюється визначення функцій, які повинна виконувати розробляється програма, а також концептуалізація предметної області. Цю роботу виконують аналітики спільно з фахівцями предметної області. Результатом даного етапу повинна бути деяка концептуальна схема, яка містить опис основних компонентів і тих функцій, які вони повинні виконувати.

Етап проектування структури програми полягає в розробці детальної схеми майбутньої програми, на якій вказуються класи, їх властивості та методи, а також різні взаємозв'язку між ними. Як правило, на цьому етапі можуть брати участь в роботі аналітики, архітектори і окремі кваліфіковані програмісти. Результатом даного етапу повинна стати деталізована схема програми, на якій вказуються всі класи і взаємозв'язку між ними в процесі функціонування програми. Згідно з методологією ООАП, саме дана схема повинна "служити вихідною інформацією для написання програмного коду.

Етап програмування навряд чи потребує уточнення, оскільки є найбільш традиційним для програмістів. Поява інструментаріїв швидкої розробки

додатків (Rapid Application Development, RAD) дозволило істотно скоротити час, і витрати на виконання цього етапу. Результатом даного етапу є програмний додаток, який володіє необхідною функціональністю і здатне вирішувати потрібні завдання в конкретній предметній області.

Етапи впровадження та супроводу програми пов'язані з необхідністю налаштування і конфігурації середовища програми, а також з усуненням виникли в процесі її використання помилок. Іноді в якості окремого етапу виділяють тестування програми, під яким розуміють перевірку працездатності програми на деякій сукупності вихідних даних або при деяких спеціальних режимах експлуатації. Результатом цих етапів є підвищення надійності програмного додатка, що виключає виникнення критичних ситуацій або нанесення збитку компанії, що використовує цю програму.

Методологія ООАП тісно пов'язана з концепцією автоматизованої розробки програмного забезпечення (Computer Aided Software Engineering, CASE). Поява перших CASE-засобів було зустрінуте з певною настороженістю. Згодом з'явилися як захоплені Відгуки про їх застосування, так і критичні оцінки їх можливостей. Причин для таких суперечливих думок було кілька. Перша з них полягає в тому, що ранні CASE-засобу були простий надбудовою над деякою системою управління базами даних (СКБД). Хоча візуалізація процесу розробки концептуальної схеми БД має важливе значення, вона не вирішує проблем розробки додатків інших типів.

Друга причина має більш складну природу, оскільки пов'язана з графічної нотації, реалізованої в тому чи іншому CASE-засобі. Якщо мови програмування мають строгий синтаксис, то спроби запропонувати відповідний синтаксис для візуального представлення концептуальних схем БД були сприйняті далеко неоднозначно. З'явилося кілька підходів, які більш детально будуть розглянуті в розділі 2. На цьому тлі поява уніфікованої мови моделювання (Unified Modeling Language, UML), який орієнтований на

вирішення завдань перших двох етапів ЖЦ програм, було сприйнято з великим оптимізмом вся спільнота корпоративних програмістів.

Останнє, на що слід звернути увагу, це усвідомлення необхідності побудови попередньої моделі програмної системи, яку, відповідно до сучасних концепцій ООАП, слід вважати результатом перших етапів ЖЦ програми. Оскільки мова UML навіть у своїй назві має відношення до моделювання, слід додатково зупинитися на цілій низці досить важливих питань. Таким чином, ми переходимо до теми, яка традиційно не розглядається в виданнях по ООАП, однак вона має саме пряме відношення до процесу побудови моделей і, власне, моделювання. Йдеться про методологію системного аналізу і системного моделювання.

5.4. Методологія системного аналізу та системного моделювання

Системний аналіз як наукова дисципліна має давнішу історію, ніж ООП і ООАП, і власний предмет дослідження. Центральним поняттям системного аналізу є поняття системи, під якою розуміється сукупність об'єктів, компонентів або елементів довільної природи, що утворюють певну цілісність. Визначальною передумовою виділення деякої сукупності як системи є виникнення у неї нових властивостей, яких не мають складові її елементи. Прикладів систем можна привести досить багато - це персональний комп'ютер, автомобіль, людина, біосфера, програма та ін. Більш ортодоксальна точка зору передбачає, що всі навколишні нас предмети є системами.

Найважливішими характеристиками будь-якої системи є її структура і процес функціонування. Під структурою системи розуміють стійку в часі сукупність взаємозв'язків між її елементами або компонентами. Саме структура пов'язує воєдино всі елементи і перешкоджає розпаду системи на окремі компоненти. Структура системи може відображати найрізноманітніші взаємозв'язку, в тому числі і вкладеність елементів однієї системи в іншу. У

цьому випадку прийнято називати більш дрібну або вкладену систему підсистемою, а більшу - метасистема.

Процес функціонування системи тісно пов'язаний зі зміною її властивостей або поведінки в часі. При цьому важливою характеристикою системи є її стан, під яким розуміється сукупність властивостей або ознак, які в кожен момент часу відображають найбільш суттєві особливості поведінки системи.

Розглянемо наступний приклад. В якості системи уявімо собі "Автомобіль". Для цього випадку система охолодження двигуна буде підсистемою "Автомобіля". З одного боку, двигун є елементом системи "Автомобіль". З іншого боку, двигун сам є системою, що складається з окремих компонентів, таких як циліндри, свічки запалювання та ін. Тому система "Двигун" також буде підсистемою системи "Автомобіль".

Структура системи "Автомобіль" може бути описана з різних точок зору. Найбільш загальне уявлення про структуру цієї системи дає механічна схема пристрою того чи іншого автомобіля. Взаємодія елементів в цьому випадку носить механічний характер. Стан автомобіля можна розглядати також з різних точок зору, найбільш загальною з яких є характеристика автомобіля як справного або несправного. Очевидно, що кожне з цих станів в окремих ситуаціях може бути деталізовано. Наприклад, стан "несправний" може бути конкретизовано в стану "несправність двигуна", "несправність акумулятора", "відсутність подачі палива" та ін. Важливо мати чітке уявлення, що подібна деталізація повинна бути адекватна розв'язуваної задачі.

Процес функціонування системи відображає поведінку системи в часі і може бути представлений як послідовна зміна її станів: Якщо система змінює одне свій стан на інше, то прийнято говорити, що система переходить з одного стану в інший. Сукупність ознак або умов зміни станів системи в цьому випадку називається переходом. Для системи з дискретними станами процес функціонування може бути представлений у вигляді послідовності

станів з відповідними переходами. Точніше графічне опис процесу функціонування систем буде дано в розділі 2.

Методологія системного аналізу служить концептуальною основою системно-орієнтованої декомпозиції предметної області. В цьому випадку вихідними компонентами концептуалізації є системи і взаємозв'язки між ними. При цьому поняття системи є більш загальним, ніж поняття класів і об'єктів в ООАП. Результатом системного аналізу є побудова деякої моделі системи або предметної області.

Поняття моделі настільки широко використовується в повсякденному житті, що набуло дуже багато смислових відтінків. Це і "Будинок моделей" відомого кутюр'є, і модель престижної марки автомобіля, і модель політичного керівництва, і математична модель коливань маятника. Стосовно до програмних систем нас буде цікавити тільки те поняття моделі, яке використовується в системному аналізі. А саме, під моделлю будемо розуміти деяке уявлення про систему, що відображає найбільш істотні закономірності її структури і процесу функціонування і зафіксоване на деякій мові або в іншій формі.

Прикладів моделей можна привести досить багато. Наприклад, аеродинамічна модель гоночного автомобіля або проектованого літака, модель ракетного двигуна, модель коливальної .Системи, модель системи електропостачання регіону, модель виборчої компанії і ін.

Загальною властивістю всіх моделей є їх подібність оригінальній системі або системі-оригіналу. Важливість побудови моделей полягає в можливості їх використання для отримання інформації про властивості або поведінці системи-оригіналу. При цьому процес побудови і подальшого застосування моделей для отримання інформації про систему-оригіналі отримав назву моделювання.

Найбільш загальною моделлю системи є так звана модель "чорного ящика". В цьому випадку система представляється у вигляді прямокутника, внутрішній устрій якого приховано від аналітика або невідомо. Однак

система не є повністю ізольованою від зовнішнього середовища, оскільки остання надає на систему деякі інформаційні або матеріальні впливу. Такі дії отримали назву вхідних впливів. У свою чергу, система також надає на середу або інші системи певні інформаційні або матеріальні впливу, які отримали назву вихідних впливів. Графічно дана модель може бути зображена наступним чином (рис. 5.7).

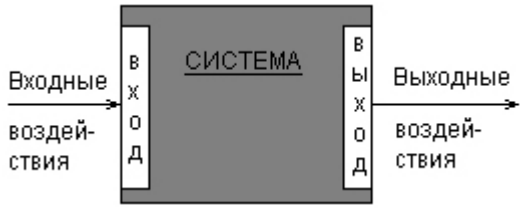

Рис. 5.7. Графічне зображення моделі системи у вигляді "чорного ящика"

Цінність моделей, подібних моделі "чорного ящика", вельми умовна. Мимоволі може виникнути асоціація з "Чорним квадратом". Однак якщо оцінка образотворчих особливостей останнього не входить в завдання системного аналізу, то загальна модель системи містить деяку важливу інформацію про цей функціональні особливості даної системи, які дають уявлення про її поведінці. Дійсно, крім самої загальної інформації про те, на які впливи реагує система, і як проявляється ця реакція на навколишні об'єкти і системи, іншої інформації ми отримати не можемо. В рамках системного аналізу розроблені певні методологічні засоби, що дозволяють виконати подальшу конкретизацію загальної моделі системи.

Процес розробки адекватних моделей та їх подальшого конструктивного застосування вимагає не тільки знання загальної методології системного аналізу, але і наявності відповідних образотворчих засобів або мов для фіксації результатів моделювання та їх документування. Очевидно, що природна мова не цілком підходить для цієї мети, оскільки володіє неоднозначністю і невизначеністю. Для побудови моделей були розроблені досить серйозні теоретичні методи, засновані на розвитку математичних і

логічних засобів моделювання, а також запропоновані різні формальні і графічні нотації, що відображають специфіку вирішуваних завдань. Важливо представляти, що уніфікація будь-якої мови моделювання тісно пов'язана з методологією системного моделювання, т. ч. складність системи і, відповідно, її моделі може бути розглянута з різних точок зору. Перш за все, можна виділити складність структури системи, яка характеризується кількістю елементів системи і різними типами взаємозв'язків між цими елементами. Якщо кількість елементів перевищує деяке порогове значення, яке не є строго фіксованим, то така система може бути названа складною. Наприклад, якщо програмна СУБД налічує більше 100 окремих форм введення і виведення інформації, то багато програмістів вважатимуть її складною. Транспортна система сучасних мегаполісів також може служити прикладом складної системи.

Другим аспектом складності є складність процесу функціонування системи. Це може бути пов'язано як з непередбачуваним характером поведінки системи, так і неможливістю формального уявлення правил перетворення вхідних впливів у вихідні. Як приклади складних програмних систем можна привести сучасні операційні системи, яким властиві риси складності як структури, так і поведінки.

6. ОСНОВИ ОБ'ЄКТНО-ОРІЄНТОВАНОГО МОДЕЛЮВАННЯ

6.1. Класифікація програмних систем

Програмна система (або *програмний продукт*) – організована сукупність програм і/або програмних модулів постійного застосування для розв'язування задач у різноманітних сферах людської діяльності.

Програмний модуль (або просто *модуль*) – частина програми, оформлена у вигляді, який допускає її незалежну компіляцію і використання. Програма – організована сукупність модулів, серед яких один з модулів є *головним*. Головний модуль організовує ро- боту інших модулів програми і забезпечує інтерфейс користувача.

Програмна система може складатися з однієї чи декількох програм (комплексу програм), які певним чином організовані та взаємодіють між собою. Програми комплексу можуть використовувати певний набір модулів (бібліотеку модулів). Модулі та їхні набори у різних мовах програмування організовані по-різному.

Термін "програмна система" узагальнений, отож вводять інші подібні терміни, які враховують область застосування чи мас- штаб використання програмної системи.

Програмна система за *областю застосування* може бути:

- *прикладною програмою* (синоніми: застосування, додаток, аплікація – від *application*), призначеною для розв'язування певного класу однотипних задач і отримання конкретних результатів (наприклад, нарахування заробітної плати);
- *пакетом* (чи *системою*) *програм*, призначених для підтримання певної професійної діяльності людини (наприклад, математичні пакети, графічні пакети, видавничі системи, системи мультимедіа тощо);
- *системною програмою* (синонім: програмне забезпечення), призначеною для забезпечення роботи прикладних програм (наприклад,

операційна система);

- *системою керування базами даних* (СКБД) – комплексом про- грам і мовних засобів, призначених для створення, ведення і використання *баз даних* (БД);
- *застосуванням реального часу* – застосуванням, в якому врахування реального астрономічного часу є критично важливим для виконання головних функцій (наприклад, диспетчерські системи на транспорті);
- *вертикальним застосуванням* – застосуванням, зробленим за конкретним індивідуальним замовленням;
- *горизонтальним застосуванням* – застосуванням, розрахованим на масовий попит, яке можна придбати у торговельній мережі (наприклад, комп'ютерні ігри);
- *офісним застосування* – застосуванням (вертикальним чи горизонтальним), призначеним для введення/виведення, зберігання та опрацювання документів у рамках організації чи підприємства, яке не зв'язане критично з часом;
- *застосування баз даних* – офісне застосування, головна функціональність якого полягає у забезпеченні зручного доступу до бази даних.

За масштабом використання програмної системи вирізняють:

- *настільні застосування* (для роботи одного користувача);
- *групові застосування* (для роботи однієї категорії користувачів локальної мережі);
- *корпоративні програмні системи* (для роботи різних категорій користувачів локальної чи глобальної мережі).

6.2. Життєвий цикл програмних систем

Програмні системи за час існування зазнають найрізноманітніших змін своєї форми, які залежать від стану процесу розробки та експлуатації

застосування. Послідовність цих змін позначають терміном *життєвий цикл*. З цим терміном тісно переплітається поняття *процесу розробки* застосування.

У буденному використанні "*розробка*" означає створення "*чогось*", розробку завершено. Після розробки розпочинається використання "чогось" (*експлуатація*). Для програмних застосувань чітко відокремити фази розробки та експлуатації не вдається. Отож тут доцільніше послуговуватись терміном "*розробка*, *що продовжується*", який поєднує в собі як термін "розробка" у звичному розумінні, так і модифікацію програми у процесі експлуатації.

Отже, терміни "*життєвий цикл*" і "*процес розробки*" можна вважати двома різними сторонами одного поняття. При вживанні терміну "життєвий цикл" мають на увазі погляд з точки зору про- грами, а при вживанні терміну "процес розробки" – погляд з точки зору програміста.

Відповідно до стандарту ISO/IEC 12207 життєвий цикл програми налічує такі етапи:

- аналіз предметної області і формулювання системних вимог (постановка задачі);
- проектування структури програми;
- реалізація програми в кодах (власне програмування);
- впровадження програми і тестування;
- супровід програми під час експлуатації;
- відмовлення від використання програми.

На *етапі аналізу предметної області і формулювання вимог* здійснюється визначення функцій, що повинна виконувати про- грама, а також концептуалізація (визначення об'єктів) предметної області. Цю роботу виконують аналітики спільно з фахівцями предметної області. Результатом є деяка *концептуальна схема*, що містить опис базових компонентів (об'єктів) і тих функцій, які вони виконуватимуть.

Здебільшого, на цьому етапі формується *словник* (*глосарій*) предметної області, який містить текстовий опис термінів, сутностей, користувачів тощо,

а також формується *технічне завдання*, в якому подано функціональні та нефункціональні вимоги до системи.

Етап *проектування* структури програми полягає в розробці детальної схеми, на якій зазначено класи, їхні властивості і методи, а також різні взаємозв'язки між ними. Зазвичай, на цьому етапі у роботі можуть брати участь аналітики, архітектори системи, а також окремі кваліфіковані програмісти.

Результатом цього етапу повинна стати деталізована схема програми, за якою описано всі класи і взаємозв'язки між ними в процесі функціонування програми.

Етап *програмування* є найтрадиційнішим для програмістів. Поява інструментаріїв швидкої розробки додатків (Rapid Application Development, RAD) дає змогу істотно скоротити час і ви- трати на виконання цього етапу. Результатом такого етапу є програмний додаток, що має необхідну функціональність і здатний вирішувати потрібні задачі у конкретній предметній області.

Етапи *впровадження* і *супроводу* програми зв'язані з необхідністю настроювання і конфігурації середовища програми, а також з усуненням помилок, які виникли під час її використання. Іноді окремим етапом вважають *тестування* програми, під яким розуміють перевірку працездатності програми на деякій сукупності вихідних даних чи за деяких спеціальних режимів експлуатації. Результатом цих етапів є підвищення надійності додатка, що виключає виникнення критичних ситуацій.

Життєвий цикл програми налічує певні етапи, які циклічно повторюються. На кожному етапі виникають або модифікуються певні матеріали (*артефакти*), які використовують на наступному етапі як вхідні дані. На рис. 6.1 назви артефактів підкреслено.

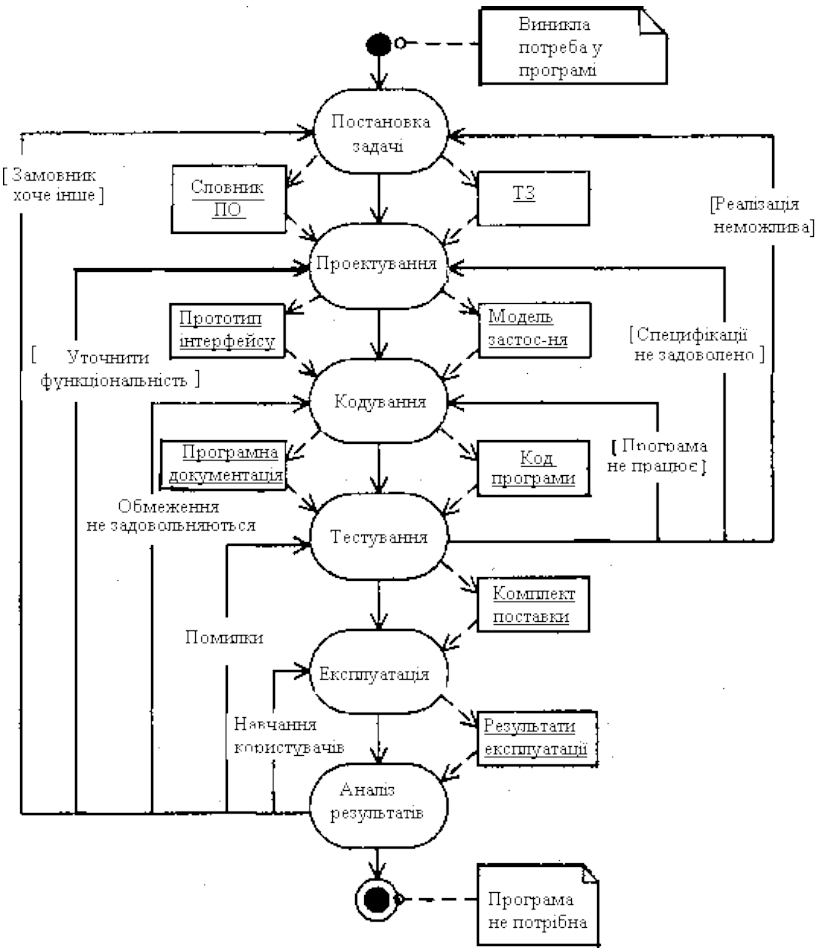

Рис. 6.1. Життєвий цикл програми

Розглядаючи різні етапи життєвого циклу програми, необхідно зазначити: якщо виникнення RAD-інструментаріїв дає змогу істотно скоротити терміни етапу програмування, то відсутність від- повідних засобів для перших двох етапів тривалий час стримувала процес розробки додатків. Розвиток методології *об'єктно-орієнтованого аналізу і проектування* (ООА і П) програмних систем було спрямовано на автоматизацію другого, а потім і першого етапів життєвого циклу програми.

Методологія ООА і П тісно переплітається з концепцією автоматизованої розробки програмних систем (Computer Aided Software Engineering, CASE). До перших CASE-засобів ставилися з певною недовірою. Згодом з'явилися як захоплені відгуки про їхнє застосування, так і критичні оцінки їхніх можливостей. Причин для таких суперечливих думок було декілька. Перша з них полягає у тому, що ранні CASE-засоби були простою

надбудовою над деякою системою керування базами даних. Хоча візуалізація процесу розробки концептуальної схеми бази даних має велике значення, проте вона не вирішує проблем розробки додатків інших типів.

Друга причина має складнішу природу, оскільки зв'язана з графічною нотацією, реалізованою у CASE-засобі. Якщо мови програмування мають строгий синтаксис, то спроби запропонувати придатний синтаксис для візуального зображення концептуальних схем БД сприйняли не надто прихильно. Отож розробку і стандартизацію *уніфікованої мови моделювання* (Unified Modeling Language, UML), орієнтованої на об'єктно-орієнтований аналіз і проектування програмних систем, співтовариство корпоративних програмістів сприйняло з великим оптимізмом.

У кожної організації, яка спеціалізується на розробці програмних систем, зазвичай існують власні моделі життєвого циклу та процесу розробки програми. Однак внаслідок широкого використання Case-засобів останніми роками набувають поширення *універсальні* та певним чином *стандартизовані* моделі.

Найпоширенішими сьогодні є моделі MS Solution Framework (MSF) корпорації Microsoft та Unified Software Development Process (USDP) фірми Rational. Модель життєвого циклу рис. 1.1 слугує певним узагальненням і спрощенням цих двох моделей.

6.3. Вступ у процес моделювання

Компанії, які займаються випуском програмних систем, досягають успіху у випадку, коли їхня продукція має високу якість і максимально враховує запити користувачів.

Для швидкої та ефективної розробки програмного продукту необхідно залучити кваліфіковану робочу силу, вибрати правильні інструменти і визначити правильний напрям роботи. Процес розробки проекту необхідно добре продумати, щоб швидко адаптувати його до можливих змін вимог

користувачів, потреб бізнесу чи технології. Цього можна досягнути, якщо у процесі розробки проекту використовувати *моделі*.

Безперечно, розробка сучасних програмних продуктів не- можлива без попереднього моделювання. Накопичений досвід засвідчує: чим більшим і складнішим є проект, тим важливішим стає *моделювання* майбутньої системи. Не варто сподіватися на успішність проекту, якщо не приділено достатньої уваги попередньому моделюванню системи.

Моделювання використовують не тільки під час створення великих систем. Адже чим більша і складніша система, тим більшого значення набуває моделювання при її розробці. Справа у то- му, що моделювати складну систему необхідно у будь-якому ви- падку, оскільки інакше ми не зможемо її представити як єдине ціле. Моделі наочно демонструють бажану структуру та поведінку системи, відображають її архітектуру та допомагають уточнити деталі проекту з замовником для мінімізації майбутніх ризиків.

Модель – це спрощене представлення реальності, своєрідне "креслення" системи. Кожну систему можна описати по-різному, використовуючи різні моделі, кожна з яких є *семантично замкну- тою абстракцією* системи.

Моделі програмних систем відображають певні аспекти системи. *Структурна модель*, наприклад, відображає статичну організацію системи, а *модель поведінки* підкреслює динамічні процеси, притаманні системі.

Зазвичай, моделювання будь-якої системи супроводжується створенням *множини моделей* для відображення різних аспектів системи. Окрім цього, моделі можуть мати різні *рівні абстракції*, які відображають одні й ті ж аспекти системи з різним ступенем деталізації.

Моделювання дає змогу розв'язати такі задачі:

- *візуалізація системи* – візуальне відображення програмної системи у її поточному чи бажаному стані;
- *специфікація системи* – визначення структури і/або поведінки системи;
- *конструювання системи* – отримання шаблону, який допоможе сконструювати систему;

• *документування системи* – фіксація прийнятих рішень на ос- нові отриманих моделей.

При розробці програмних систем існує декілька методів моделювання, які відрізняються своєю орієнтацією на *стиль програмування.* Зазвичай, вирізняють п’ять стилів:

• процедурно-орієнтований – спрямований на представлення програми як множини процедур, які за чергою викликаються;

• об’єктно-орієнтований – спрямований на представлення про- грами як набору взаємодіючих об’єктів;

• логіко-орієнтований – спрямований на виконання цілей, які передано у термінах обчислення предикатів;

• орієнтований на правила – виконання правил “якщо-то”;

• орієнтований на обмеження.

Як стверджують автори [1], неможливо визнати один стиль програмування найкращим у *всіх* областях практичного застосування, однак об’єктно-орієнтований стиль найприйнятніший для най- ширшого кола задач.

Алгоритмічний метод моделювання передбачає процедурно-орієнтований стиль програмування, у рамках якого програміст створює процедури, що викликають одна одну для виконання поставлених задач і обробки даних.

Головним будівельним блоком є процедура чи функція, а увага приділяється насамперед питанням передачі керування і декомпозиції великих алгоритмів на менші. Нічого поганого у цьому немає, якщо не зважати на те, що системи важко адаптуються при зміні вимог чи збільшенні розміру додатка.

За використання об’єктно-орієнтованого стилю програміст створює програмні об’єкти і наділяє їх визначеною поведінкою, реакцією на зміни зовнішніх умов. Такі об’єкти взаємодіють між собою, виконують визначені задачі, приймають, обробляють і передають дані.

На об’єктно-орієнтованому програмуванні базується об’єкт- но-

орієнтований метод моделювання. Якщо об’єктно-орієнтоване програмування спрямоване на правильне й ефективне використання об’єктів у рамках конкретних мов, то об’єктно-орієнтоване моделювання спрямоване на правильне й ефективне структурування складних систем.

Об’єктно-орієнтований метод моделювання передбачає такий *аналіз* предметної області, за якого уже на початкових етапах розробки програмної системи можна було б вирізняти набори взаємо- діючих об’єктів.

Об’єктно-орієнтована модель має чотири головні властивості:

- *абстрагування* – виокремлення істотних характеристик об’єкта, що вирізняють його з-поміж інших видів об’єктів;
- *інкапсуляція* – приховування внутрішньої реалізації об’єкта за наданим цим об’єктом інтерфейсом;
- *модульність* – здатність системи розкладатися на внутрішньо сильно чи слабко зв’язані між собою модулі;
- *ієрархія* – упорядкування абстракцій і розташування їх за рівнями.

Ці властивості є головними, і за відсутності будь-якого з них модель не буде об’єктно-орієнтованою.

Існує також три додаткових властивості, корисні в об’єктній моделі, без яких, однак, можна обійтися:

- *типізація* – створення об’єктів на основі шаблонів визначеного типу;
- *паралелізм* – здатність системи обробляти декілька повідомлень чи задач паралельно;
- *збережуваність* – здатність системи зберігати не тільки дані, але й об’єкти у проміжку між окремими запусками системи.

Об’єктно-орієнтоване моделювання програмних систем довело свою корисність при побудові систем будь-якого розміру і складності у різних областях людської діяльності. Окрім того, сучасні мови програмування, інструментальні засоби та операційні системи, здебільшого, є тією чи іншою мірою об’єктно-орієнтованими, а це дає вагомі підстави трактувати світ у термінах об’єктів.

6.4. Класи та об’єкти

Для створення об’єктно-орієнтованої програми необхідно створити деякий набір об’єктів з визначеною поведінкою та схемою їхнього взаємозв’язку. У свою чергу, для створення об’єктів необхідно попередньо створити їхній опис, який у термінах C++ називають класом.

Клас – це абстракція сукупності об’єктів, які мають спільний набір властивостей і володіють однаковою поведінкою. Об’єктом називають екземпляр відповідного класу. Об’єкти, які не мають ідентичних властивостей чи не володіють однаковою поведінкою, за визначенням не можуть належати одному класу.

Клас – це шаблон, на основі якого створено об’єкти. Не можна плутати клас і об’єкт. Клас – це лише матриця, на основі якої створюють об’єкти.

Клас визначеного типу може бути тільки один, а об’єктів у програмі може бути скільки завгодно (точніше, наскільки вистачить ресурсів системи). Перевірити правильність опису створеного класу можна тільки після створення об’єктів на його основі. Коли об’єкти починають працювати і взаємодіяти, тільки тоді можна оцінити точність поводження об’єкта, описаного класом.

Якщо взяти приклад телефонного апарата, то його електрон- на схема – те саме, що клас, а сам апарат – те саме, що об’єкт. На основі однієї електронної схеми можна зробити скільки завгодно апаратів, і усі вони працюватимуть і виглядатимуть однаково за умови, що на заводі не допущено браку. У цьому відмінність програмування від реального виробництва. Під час створення програми за точністю виготовлення об’єкта стежить мовний компілятор, а програмістові необхідно зосередитись на правильному описі класу. Найважливішими властивостями класів вважають інкапсуляцію, успадкування і поліморфізм.

Інкапсуляція класу аналогічна властивості об’єктно-орієнтованої моделі і

має на увазі приховання непотрібних деталей реалізації класу. У самому класі можуть зберігатися дані і методи їхньої обробки, доступні тільки за допомогою наданого класом інтерфейсу і захищені від небажаного впливу ззовні. Для використання класу немає необхідності знати його внутрішню будову (адже для того, щоб скористатися телефоном, немає необхідності знати його електронну схему). Необхідно лише знати, як набрати номер чи відповісти на дзвінок, тобто знати опис зовнішнього інтерфейсу.

Успадкування – одна із найважливіших властивостей класу. Ця властивість дає змогу створювати на основі одного чи декількох *батьківських* класів *дочірні* класи (підкласи) із властивостями батьківських і власними додатковими можливостями.

Можливість успадкування властивостей дає змогу програмістові скоротити обсяг ручного кодування і змінити поведінку всіх дочірніх об’єктів унаслідок зміни поведінки батьківського класу.

Якщо в реальному виробництві у схемі уже випущених телефонних апаратів буде знайдено помилку, то всю партію доведеться викинути на смітник. У програмуванні – навпаки: виправляючи знайдену помилку в батьківському класі, ми після запуску програми змушуємо правильно працювати усі дочірні класи.

Поліморфізм – можливість об’єктів, створених на основі класів, змінювати свою реакцію на ті ж самі впливи за різних зовнішніх умов.

Класи мають атрибути і методи. У програмі *атрибути* (або *властивості*) – це змінні, описані в тілі класу, що можуть бути як сховані від зовнішнього впливу (зміна властивостей виробляється за допомогою доступних ззовні методів), так і доступні для зміни. *Методи* – це функції, визначені в тілі класу, що можуть бути доступні чи сховані від зовнішніх програм.

6.5. Методологія об’єктно-орієнтованого моделювання

Метод об’єктно-орієнтованого моделювання передбачає послідовне виконання двох етапів: об’єктно-орієнтованого *аналізу* та об’єктно-орієнтованого *проектування*. Тому термін “об’єктно-орієнтоване моделювання” еквівалентний терміну “об’єктно-орієнтований аналіз і проектування” (ООА і П).

Аналіз – широке поняття. Його зміст детальніше відображають терміни *аналіз системних вимог* (тобто дослідження вимог до майбутньої програмної системи) та *об’єктний аналіз* (тобто дослідження об’єктів предметної області).

При цьому під *предметною областю* розуміють ту частину реального світу, що має істотне значення чи безпосереднє відношення до процесу функціонування програми. Іншими слова- ми: предметна область містить у собі тільки ті об’єкти і взаємо- зв’язки між ними, які необхідні для опису вимог і умов розв’язання деякої задачі.

У загальному випадку виділення базових об’єктів (чи компонентів) предметної області є нетривіальною задачею. Складність виявляється у неформальному характері процедур чи правил, які можна застосовувати з цією метою. Окрім того, таку роботу необхідно виконувати спільно з фахівцями чи експертами, що добре знають предметну область.

У процесі *проектування* головну увагу звертають на концептуальні рішення, які забезпечують виконання системних вимог, а не на питання реалізації. У процесі об’єктно-орієнтованого проектування визначають програмні об’єкти та способи їхньої взаємодії і/або схеми баз даних.

Під час *аналізу системних вимог* необхідно з’ясувати потре- би *замовника*, аналізуючи отриману інформацію від керівництва компанії та майбутніх користувачів системи. Під час аналізу необхідно визначити:

- *функціональні вимоги* до системи (або *бізнес-процеси*), тобто встановити *варіанти використання* програмної системи для реалізацій

конкретних функцій чи дій у даній предметній області (“визначити те, що система *має робити*”);

- *потоки даних* для кожного бізнес-процесу;
- *границі* системи;
- *користувачів* системи та процеси їхньої взаємодії з системою.

У результаті аналізу, зазвичай, оформляють *словник* (або *глосарій*) предметної області (містить текстовий опис термінів, сутностей, користувачів тощо) і *технічне завдання*, у якому сформульовано функціональні та нефункціональні вимоги до системи. До *нефункціональних* вимог зачислено питання надійності, зручності використання, продуктивності, можливості супроводу програм, питання безпеки, проектні та апаратні обмеження тощо.

Технічне завдання є гарантією єдиного трактування вимог замовниками і проектувальниками. Воно дає змогу також вирішувати спірні питання з приводу функцій системи, що виникають у процесі її створення.

В UML певним синонімом терміну “*технічне завдання*” є *специфікація вимог* до системи (Software Requirement Specification, SRS), в яких визначаються межі системи, користувачів і функціональні вимоги. SRS є текстовою основою *формалізації* етапу постановки задачі за допомогою діаграм прецедентів.

7. ОСНОВИ УНІФІКОВАНОЇ МОВИ МОДЕЛЮВАННЯ (UML)

7.1. Загальна характеристика UML

Уніфікована мова моделювання (Unified Modeling Language, UML) – це графічна мова для *специфікації*, *візуалізації*, *конструювання* і *документування* програмних систем. За допомогою UML можна розробити детальний план такої системи, який відображатиме і *концептуальні* елементи системи (системні функції та бізнес-процеси), і особливості її *реалізації* (класи, схеми баз даних, програмні компоненти багаторазового використання тощо).

Авторами мови є Грейді Буч (Grady Booch), Джеймс Рамбо (James Rumbaugh) і Айвар Якобсон (Ivar Jacobson). У січні 1997 ро- ку внаслідок об'єднання розробок цих авторів випущено версію UML 1.0, а в листопаді 1997 року – версію UML 1.1. Наступні версії: UML 1.3 – квітень 1999 року; UML 1.4 – жовтень 2001 року.

Зауважимо, що базові ідеї та конструкції мови практично не змінювалися з версії UML 1.1. У наступних версіях уточнювали визначення, добавляли розділи, регламентували зв'язки UML з іншими технологіями тощо. Паралельно розвивалися інструментальні засоби, які підтримували UML (Rational Rose Enterprise, Objec- teering, Magic Draw UML, MS Visio, Visual UML та ін.).

Сьогодні UML є загальновизнаним стандартом, який використовує більшість розробників системного та прикладного програмного забезпечення. Знань UML вимагають не лише від системних аналітиків та проектувальників, але й від звичайних програмістів і тестувальників програмного забезпечення. Постійно збільшується ринок UML-орієнтованих інструментальних засобів, призначених для автоматизації процесу розробки програм.

Безперечно, UML відіграватиме важливу роль у галузі розробки

програмного забезпечення і в майбутньому. Розвиток UML буде спрямований на спрощення розв'язку однієї з найскладніших задач в галузі інформаційних технологій – задачі проектування програмного забезпечення.

UML призначено для моделювання програмних систем. Самі автори UML визначають її як графічну мову моделювання загального призначення, яку використовують для специфікації, візуалізації, конструювання і документування усіх артефактів1, які створюються під час розробки програмних систем.

- *Специфікація* – це декларативний опис того, як усе побудоване або працює. UML надає достатньо формальні та універсальні засоби для специфікації усіх можливих артефактів, що дає змогу знизити ризик неоднозначного сприйняття специфікації.

- *Візуалізація* – представлення інформації у графічній формі, придатній для сприйняття людиною. Часто моделювання є єдиним засобом, який дає змогу уявити систему загалом як одне ціле. Проблема полягає в обмеженому сприйнятті людиною складних сутностей. Моделювання передбачає розділення складної системи на дещо простіші складові та окремо розглядає кожну з цих складових. Також моделювання дає змогу створювати високорівневі моделі всієї системи, відкидаючи деталі, несуттєві для цього рівня абстракції.

- *Конструювання* – отримання набору програмних модулів, які утворюють застосування або його компонент. Розроблені моделі системи утворюють деякий базовий каркас, на основі якого можна будувати систему. Сучасні CASE-засоби дають змогу деякою мірою автоматизувати конструювання програмного ко- ду на підставі розроблених моделей.

- *Документування проектних рішень.* Для підтримки та розвитку програмних продуктів потрібна вичерпна та якісна документація. Моделювання дає змогу одержати документи, які визначають високорівневу організацію системи. Такі документи необхідні для початкового ознайомлення з системою.

Серед головних властивостей UML можна виокремити такі:

- *UML – це мова моделювання*, яку використовують в контексті деякого процесу розробки програмних засобів.
- *UML – це функціонально завершена мова*, яка забезпечує пов- ний цикл моделювання програмного забезпечення, починаючи від формування концепції майбутньої системи і завершуючи питаннями програмної реалізації системи.
- *UML – це об'єктно-орієнтована мова*, яку найефективніше можна застосовувати саме в контексті об'єктно-орієнтованих методів аналізу та проектування.
- *UML – це формальна мова*, яка дає змогу будувати завершені моделі, яким властива однозначна інтерпретація. Наслідком формалізації мови є можливість її інтерпретації не лише людьми, але й машинами.
- *UML – це мова візуалізації*, орієнтована на представлення моделей програмних систем переважно в графічній формі. Водночас UML припускає долучення до моделі текстової інформації з метою додаткової конкретизації деталей.
- *UML – це стандартна мова*, яка гарантує вільне розуміння та поширення UML-моделей між різними розробниками.
- *UML – це універсальна мова*, яка з однаковим успіхом придатна для моделювання як великих, так і малих програмних систем.
- *UML – це незалежна мова*, яка не накладає обмежень на мови програмування для реалізації моделей і може бути сумісною з будь-якою об'єктно-орієнтованою мовою.

Використання UML найефективніше в інформаційних системах масштабу підприємства, банківських і фінансових установах, телекомунікаціях, на транспорті, у торгівлі, науці тощо.

UML можна використовувати, наприклад, для моделювання документообігу в юридичних фірмах чи керівних структурах, для опису структури та функціонування системи обслуговування пацієнтів у лікарнях,

для проектування апаратних засобів тощо.

Зазначимо ще одну область, у якій активно застосовують графічну нотацію – це виконання робіт з приведення системи менеджменту якості у відповідність зі стандартом ISO 9001:2000 у рамках сертифікації підприємств і компаній. У цій області мову UML застосовують для візуального моделювання та документування бізнес- процесів. Розроблені діаграми і пояснення до них надсилають між- народним сертифікаційним органам для отримання відповідного сертифікату.

Концептуальна модель (або метамодель) UML складається з трьох частин: *архітектурного базису*, *правил мови* та *загальних механізмів мови*.

7.2. Архітектурний базис UML

Архітектурний базис UML визначає базові поняття, якими оперує мова: *сутності*, *відношення* та *діаграми*.

Сутності – це певні абстракції, які є базовими елементами моделей. В UML є чотири типи сутностей: *структурні* (актори, класи, інтерфейси, компоненти, вузли), *поведінки* (прецеденти, діяльності, стани і повідомлення), *групування* та *анотаційні*.

Структурні сутності – це статичні поняття, які відповідають концептуальним, логічним чи фізичним елементам системи. Структурні сутності, зазвичай, позначають *іменниками*. Розрізняють п’ять *головних* структурних сутностей: *актори*, *класи*, *інтерфейси*, *компоненти*, *вузли*. Кожна з сутностей може мати свої підвиди.

Актор (Actor) – це суб’єкт, який перебуває поза системою, що моделюється, і безпосередньо з нею взаємодіє. Графічно акторів зображають значком “худа людина”, під яким вказують ім’я актора (рис. 7.1).

Клас (Class) – це сукупність *однотипних сутностей* предметної області (*об’єктів*) зі спільними атрибутами, операціями, відношеннями та семантикою.

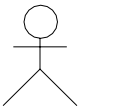

Рис. 2.1. Зображення актора

В UML класи зображають прямокутником, розділеним на три секції, в яких записують назву класу, атрибути та операції, відповідно (рис. 7.2). Назву абстрактного класу позначають *курсивом*. Атрибути та операції мають чітко визначені формати запису, які відображають їхні найважливіші характеристики (назви, типи то- що). За необхідності секції атрибутів і/або операцій опускають.

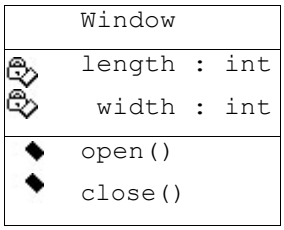

Рис. 7.2. Зображення класу

Об'єкти (Objects) – це *екземпляри класів* з конкретними значеннями атрибутів. Об'єкт має зображення, подібне до зображення класу, проте назву об'єкта підкреслюють і записують у вигляді:

<назва об'єкта>:<назва класу>.

Якщо ідентифікація об'єкта неважлива, то вказують лише назву класу, до якого належить об'єкт:

:<назва класу>.

При зображенні об'єктів секції атрибутів та операцій, здебільшого, опускають.

Інтерфейс (Interface) – це сукупність операцій, що формують деякий сервіс, який надає клас чи компонент. Інтерфейс лише декларує операції, а

реалізація операцій покладається на клас або компонент, який підтримує цей інтерфейс. Інтерфейси зображають колом, під яким вказують назву інтерфейсу (рис. 7.3).

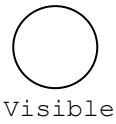

Рис. 7.3. Зображення інтерфейсу

Компонент (Component) – це фізично заміщувана частина системи, яка відповідає певному набору інтерфейсів і/або забезпечує реалізацію іншого набору інтерфейсів. Компоненти фізично існують під час виконання програми. Графічне зображення компонента – прямокутник з двома виступами з лівого боку і назвою усередині (рис. 7.4).

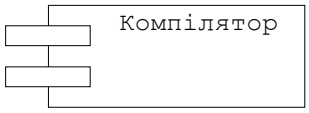

Рис. 7.4. Зображення компонента

В означенні компонента цілковито викладено його семантику:

- компонент має *фізичну* природу – він існує в реальному світі бітів, а не у світі концепцій;
- компонент *заміщуваний* – замість одного компонента можна підставити інший, якщо він відповідає тому ж самому набору інтерфейсів;
- компонент – це *частина* системи.

У багатьох аспектах компоненти подібні до класів: мають на- зви, можуть реалізувати інтерфейси, вступати у певні відношення, бути вкладеними, вступати у взаємодії. Однак між ними є суттєві розбіжності:

- класи – логічна абстракція, а компоненти – фізичні сутності

(можуть розміщуватися у *вузлах*);

- компоненти слугують фізичною упаковкою логічних сутностей (класів, кооперацій тощо), отож є на іншому рівні абстракції порівняно з класами;

• класи володіють атрибутами та операціями, а компоненти – лише операціями, доступними через їхні інтерфейси.

Можна виокремити три групи компонентів:

1.*Компоненти розгортання* (Deployment components): динамічно під'єднувані бібліотеки (DLL) і програми виконання (EXE).

2.*Компоненти – робочі продукти* (Work product components): файли з вихідними текстами програм чи даними; бази даних або таблиці баз даних; документи; виконавчі модулі із закрити- ми механізмами комунікації тощо.

3.*Компоненти виконання* (Execution components) – наслідок ро- боти системи (прикладом є об'єкт COM+, екземпляр якого створюється з DLL).

Вузол (Node) – це фізичний елемент системи, який існує під час виконання програми і представляє обчислювальний ресурс. Вузли зображають кубом, в якому вказується назва вузла (рис. 7.5). Вузол володіє певним обсягом пам'яті і, можливо, процесором.

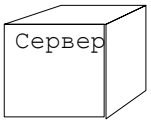

Рис. 7.5. Зображення вузла

Вузли надають засоби фізичного розгортання компонентів. Найпоширеніший приклад використання вузлів – це моделювання процесорів і пристроїв, які утворюють топологію автономної, вбудованої, клієнт-серверної чи розподіленої комп'ютерної системи.

Сутності *поведінки* (прецеденти, діяльності, стани і повідомлення) розглядатимемо під час вивчення відповідних *діаграм*.

Сутності *групування – пакети* (packages) можуть містити структурні сутності, сутності поведінки та інші сутності групування. На відміну від компонентів, які реально існують під час роботи програми, пакети мають чисто концептуальний характер (існують тільки під час процесу розробки).

Пакет зображають прямокутником із "закладкою" – меншим

прямокутником, приєднаним до верхнього лівого кута (рис. 7.6).

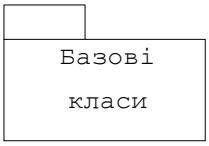

Рис. 7.6. Зображення пакета

Анотаційна сутність – це коментар для пояснення чи зауваження до будь-якого елемента моделі. Є тільки один тип анотаційної сутності – *примітка* (note). Графічно *примітку* зображають прямокутником із загнутим правим верхнім кутом (рис. 7.7).

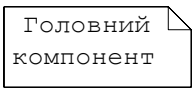

Рис. 7.7. Зображення примітки

7.3. Відношення

Відношення – це відображення семантичного зв'язку між сутностями. У мові UML визначено чотири основні типи відношення: *залежності*, *асоціації*, *узагальнення* та *реалізації*.

Залежність (dependency) – це відношення *використання*, за якого зміна однієї сутності (*незалежної*) може вплинути на іншу сутність, яка її використовує, причому зворотне використання, за- звичай, неприпустиме. Для зображення залежності використовують пунктирну лінію зі стрілкою (рис. 7.8), спрямованою у бік *незалежної* сутності.

Рис. 7.8. Зображення залежності

Найчастіше залежності використовують під час моделювання класів, щоб відобразити у сигнатурі операції той факт, що один клас використовує

інший клас (незалежну сутність) аргументом.

Асоціація (association) – це структурне відношення, що описує множину зв'язків (з'єднань) між об'єктами. Різновид асоціації – *агрегування* (aggregation) – це структурне відношення між цілим і його частинами. Графічно асоціацію зображають у вигляді лінії (іноді завершується стрілкою), поруч з якою можуть бути додаткові позначення (кратність, назви ролей тощо). На рис. 2.9 зображено приклад відношення цього вигляду.

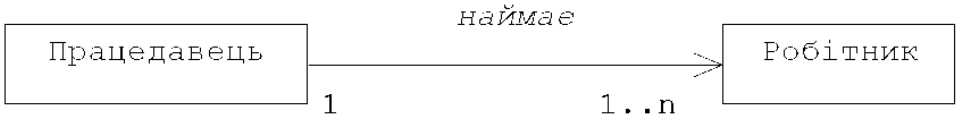

Рис. 7.9. Приклад асоціації

Узагальнення (generalization) – це відношення типу "спеціалізація/узагальнення", за якого об'єкт спеціалізованого елемента (*нащадок*) може бути підставлений замість об'єкта узагальненого елемента (*батька*, *предка*), проте не навпаки. За принципом об'єктно-орієнтованого програмування, нащадок (child) успадковує структуру і поведінку свого предка (parent). Графічно відношення узагальнення зображають лінією з незафарбованою стрілкою, яка вказує на предка (рис. 7.10).

Рис. 7.10. Приклад узагальнення

Здебільшого, у нащадка, окрім унаслідуваних, є і власні атрибути та операції. Операція нащадка з тією ж самою сигнатурою, що й у предка, заміняє відповідну операцію предка (властивість *поліморфізму*).

Найчастіше узагальнення використовують при моделюванні класів. Клас може мати одного (Single inheritance) або декілька предків (Multiple inheritance), чи не мати їх зовсім. Клас, у якого не- має предків, а є нащадки,

називають *базовим* (або *кореневим*). Клас, у якого немає нащадків, називають *листковим.*

Узагальнення використовують також з метою відображення наслідування між класами та інтерфейсами або з метою відображення наслідування між пакетами тощо.

Реалізація (realization) – це відношення між класифікаторами, за якого один класифікатор визначає зобов'язання, а інший гарантує їхнє виконання. Відношення реалізації трапляються у двох випадках:

- між інтерфейсами і класами/компонентами, що їх реалізують;
- між прецедентами і кооперaціями, що їх реалізують.

Відношення реалізації зображають у вигляді пунктирної лінії з незафарбованою стрілкою, як щось середнє між відношенням узагальнення і залежності (рис. 2.11):

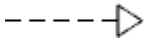

Рис. 7.11. Зображення реалізації

Ми розглянули чотири типи відношень, які є базовими у моделях UML. Існують також їхні варіанти: *уточнення* (refinement), *трасування* (trace), *долучення* і *розширення* для залежностей тощо.

7.4. Діаграми UML

Діаграма UML – це графічне зображення елементів системи у формі зв'язаного графа з *вершинами* (сутностями) і *ребрами* (відношеннями). Діаграми можуть містити будь-яку комбінацію сутностей, однак у практиці моделювання застосовують порівняно невелику кількість типових комбінацій, а саме:

- *Діаграми класів* (class diagram) – зображають класи, інтерфейси, об'єкти і кооперації, а також відношення між ними. Ці діаграми

відображають *статичні* аспекти системи.

- *Діаграми об'єктів* (object diagram) – зображають об'єкти і від- ношення між ними. Це *статичні* знімки екземплярів сутностей, зображених на діаграмах класів.

- *Діаграми прецедентів* (use case diagram) – зображають прецеденти й акторів, а також відношення між ними. Ці діаграми відображають *статичні* аспекти системи.

- *Діаграми взаємодії* – зображають об'єкти та повідомлення, якими об'єкти можуть обмінюватися. Зазвичай, розглядають два часткові випадки таких діаграм: *діаграми послідовностей* (sequence diagram), що відображають *часову* упорядкованість повідомлень, і *діаграми кооперації* (collaboration diagram), на яких зображають структурну організацію об'єктів, що обмінюються повідомленнями. Ці типи діаграм є ізоморфними. Діаграми взаємодії відображають *динамічні* аспекти системи.

- *Діаграми станів* (statechart diagram) – зображають автомат, що налічує стани, переходи, події і види дій. Ці діаграми відображають *динамічні* аспекти системи.

- *Діаграми діяльностей* (activity diagram) – це окремий випадок діаграми станів (на діаграмі зображають переходи потоку керування від одного виду діяльності до іншого усередині системи).

- *Діаграми компонентів* (component diagram) – зображають сукупність компонентів та існуючі між ними залежності. Ці діаграми відображають *статичні* аспекти системи.

- *Діаграми розміщення* (deployment diagram) – зображають конфігурацію вузлів системи і розміщених у них компонентів. Ці діаграми відображають *статичні* аспекти системи. Вони зв'язані з діаграмами компонентів, оскільки у вузлі розміщують один чи декілька компонентів.

Програмна система (або деяка інша система) – це сутність аналізу та проектування, яку розглядають з різних позицій за допомогою моделей, які відображають діаграмами. Діаграма – графічна проекція елементів системи.

Один елемент можна зображати на різних діаграмах. Кожна діаграма відображатиме одне з можливих представлень елементів системи.

7.5. Правила і загальні механізми мови UML

Елементи мови UML комбінуються один з одним за певними семантичними *правилами*, які дають змогу коректно та однозначно визначати:

- *назви*, які можна надавати сутностям, відношенням і діаграмам;
- *область дії* – контекст, в якому назва має певне значення;
- *видимість*– контекст, в якому назва може використовуватися іншими елементами;
- *цілісність* – узгоджена поведінка та взаємодія елементів;
- *виконання* – правильне розуміння поведінки системи в динаміці.

Моделювання спрощується і здійснюється ефективніше, якщо дотримуватися деяких угод. Роботу з UML істотно полегшує послідовне використання загальних механізмів: *специфікації* (specifications), *доповнення* (adornments), *поділу* (common divisions) і *розширення* (extensibility mechanisms).

Щодо кожного з елементів графічної моделі UML складається *специфікація*, яка містить текстове представлення елемента.

Наприклад, піктограмі класу відповідає специфікація, що цілковито описує його атрибути, операції (з вичерпними сигнатурами) і поведінку, а також може містити й інші деталі: видимість атрибутів і операцій, коментар про те, що клас є абстрактним тощо. Візуально ж піктограма класу відображає його найважливіші аспекти: назву, атрибути й операції.

Отже, графічну нотацію UML використовують для візуалізації системи, а за допомогою специфікацій описують її деталі. Специфікації UML визначають *семантичний план*, що містить у собі складові частини усіх моделей системи і забезпечує їхнє узгодження між собою.

Практично кожен з елементів моделі має унікальне графічне зображення, на якому відображено найважливіші аспекти цього елемента. Однак в моделі можуть використовуватися додаткові елементи зображення цього елемента, які *доповнюють* характеристику цього елемента іншими аспектами. Як головні, так і доповнюючі аспекти елемента моделі описано у специфікації UML.

Доповнення (adornments) – це текстові або графічні об'єкти, долучені до базової нотації елемента, які застосовують для візуалізації деталей його специфікації.

Наприклад, базова нотація асоціації – це лінія, однак її можна доповнювати стрілками, кратностями і назвами ролей.

При роботі з класами, компонентами чи вузлами уточнюючу інформацію можна розмістити у *додатковому* розділі, який розташовують нижче від головних розділів. З метою однозначного розуміння додатковий розділ рекомендують називати явно.

Під час моделювання об'єктно-орієнтованих систем існує *поділ* на *класи*/*об'єкти* та *інтерфейс*/*реалізацію*.

Клас – це абстракція, а *об'єкт* – конкретне втілення цієї абстракції (або екземпляр класу). Наприклад, є прецеденти й екземпляри прецедентів, компоненти й екземпляри компонентів, вузли й екземпляри вузлів тощо. У графічному зображенні об'єкта прийнято його назву підкреслювати.

Інтерфейс декларує зобов'язання, а *реалізація* представляє конкретне втілення цих зобов'язань і точно відповідає оголошеній семантиці інтерфейсу. Майже всі конструкції UML характеризуються дихотомією *інтерфейс/реалізація*. Наприклад, прецеденти реалізуються кооперaціями, а операції – методами.

UML – це стандартна мова розробки моделей програмних систем, однак жодна замкнута мова не в змозі охопити нюанси всіх можливих моделей у різних предметних областях. Отож UML є *відкритою* мовою (допускає контрольовані *розширення*). Механізми розширення UML налічують:

• *стереотипи* (stereotype) – розширюють словник UML шляхом створення нових сутностей мови на основі існуючих сутностей;

• *позначені значення* (tagged value) – розширюють властивості основних елементів UML, даючи змогу долучати нову інформацію до специфікації елемента;

• *обмеження* (constraints) – розширюють семантику елементів, даючи змогу створювати нові і скасовувати існуючі правила.

В UML чотири головні типи сутностей дають змогу моделювати величезну кількість систем. Однак іноді доцільно вводити нові сутності, специфічні для предметної області, що моделюється.

Стереотип – це деякий метаклас, який дає змогу вводити нові сутності. У найпростішому випадку стереотип зображають як назву в типографічних лапках (наприклад, «`global`»). Для наочності стереотипу можна призначити піктограму (розмістити її праворуч від назви) чи застосувати новий графічний символ. Усі три підходи проілюстровано на рис.7.12.

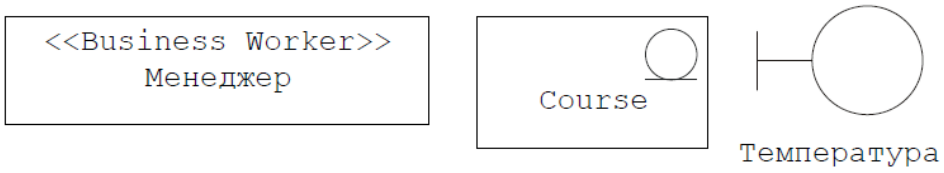

Рис. 7.12. Способи зображення стереотипів

У кожної сутності є фіксований набір властивостей: класи мають назви, атрибути й операції; асоціації –- назви й кінцеві точки (кожна зі своїми властивостями) і т. д. Позначені значення дають змогу додавати нові властивості.

Позначене значення (або *позначка*) – це *метадані* (його значення застосовують щодо сутності), яке зображають так:

```
{ [ НазваПозначки = ] Значення }
```

Фігурні дужки – обов'язкові елементи позначки. Можна вказувати

тільки значення, якщо воно допускає однозначну інтерпретацію. Позначки можна визначати для існуючих сутностей UML або застосовувати щодо окремих стереотипів. Позначки розташовують під назвою сутності чи стереотипу. У фігурних дужках може бути декілька позначених значень, які розділяються комами. На рис. 2.13 наведено приклад використання позначених значень у варіанті використання.

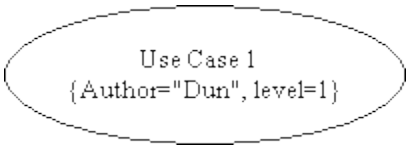

Рис. 7.13. Приклад використання позначених значень

Обмеження – це логічне твердження щодо значень властивостей елементів моделі. Задаючи обмеження, ми тим самим розширюємо семантику елементів.

Обмеження зображають рядком у фігурних дужках, який розташовують після назви елемента чи в окремій примітці, приєднаній до відповідного елемента. Залежно від інструментальних засобів рядок може бути неформальним текстом, логічним виразом мови програмування чи виразом на іншій формальній мові. Обмеження можна приєднувати відразу до декількох елементів. На рис. 7.14 наведено приклад використання обмеження.

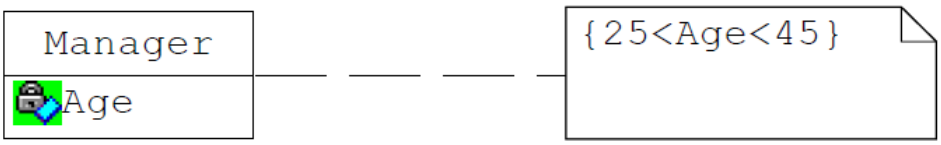

Рис. 7.14. Приклад використання обмеження

Примітка. Використання для позначок та обмежень однієї і тієї ж синтаксичної конструкції (фігурних дужок) не є випадковим. Позначене значення можна вважати обмеженням, яке може набувати тільки одного

конкретного значення.

Визначення стереотипу здійснюють так. Беруть за основу де- який існуючий елемент моделі і до нього додають нові позначені значення, нові обмеження і доповнення. Після того, як стереотип визначено, його можна використовувати як елемент моделі.

В UML визначено кілька десятків *стандартних стереотипів*, які доволі часто застосовують під час моделювання. Наприклад, *актор* не є самостійною сутністю метамоделі – це стереотип класу. Ми не описуватимемо усі стандартні стереотипи – чимало з них вам ніколи не знадобиться. Деякі стандартні стереотипи розглянемо при подальшому викладі матеріалу.

7.6. Представлення моделі

Для моделювання програмних систем необхідно розглядати їх з різних позицій. Усі, хто має відношення до проекту, – кінцеві користувачі, аналітики, прикладні програмісти, системні адміністратори, тестувальники, менеджери проектів тощо – переслідують власні інтереси, і кожен дивиться на створювану систему по- різному в різні моменти її життя.

Абстрактний граф моделі, який складається з множини різнотипних сутностей і відношень, не підлягає конструюванню загалом. З нього виокремлюють тільки сутності та відношення, акту- альні для певної точки зору (*представлення*).

Набір представлень моделі є ще менш формальним і догматичним, ніж набір канонічних типів діаграм. Найпопулярнішим вважають набір представлень, описаних авторами UML:

- *Представлення прецедентів* (Use case view) – це опис поводження системи з позиції зовнішніх користувачів, аналітиків і тестувальників.
- *Логічне представлення* (Logical view) – це опис системи з позиції проектування. Він охоплює класи, інтерфейси та кооперації, що формують

словник предметної області та її розв’язок.

- *Представлення процесів* (Process view) – це опис взаємодії процесів під час роботи системи. Він віддзеркалює такі аспекти, як паралелізм, синхронізація, продуктивність, масштабованість і пропускна здатність.
- *Представлення компонентів* (Component view) – це опис конфігурації системи з позиції реалізації, який охоплює компоненти і файли, що використовуються для збирання і випуску готового програмного продукту.
- *Представлення розміщення* (Deployment view) віддзеркалює топологію зв’язків апаратних засобів і розміщення на них компонентів.

ЛІТЕРАТУРА

1. Молчанов А.А. Моделирование и проектирование сложных систем.— К.: Вища школа, 1988.— 317с.

2. Ризкин И.Х. Машинный анализ и проектирование технических систем.— М.: Наука, 1985.— 160, с.

3. Одрин В.М., Картавов С.С. Морфологический анализ сис тем.— К.: Наукова думка, 1977.— 148 с.

4. Понтрягин Л.С., Болтянский ВТ., Гамкрелидзе Р.В., Мищенко Е.В. Математическая теория оптимальных процессов.— М.: Наука, 1976.—392 с.

5. Васильев Ф.П. Численные методы решения экстремальных задач.—М.: Наука, 1980.

6. Колесов Ю.Б., Сениченков Ю.Б.. Визуальное моделирование сложных динамических систем. Изд. «Мир и Семья & Интерлайн», СПб, 2000, 242с.

7. Боггс У, Боггс М. UML и Rational Rose, М.: Лори, 2000. - 582с.

8. Буч Г., Рамбо Д., Джекобсон А. Язык UML. Руководство пользователя: Пер. с англ. – М.: ДМК, 2000. – 432с.

9. Ledin J. Simulation Engineering. CMP Books, Lawrence, Kansas, 2001.