**Міністерство освіти і науки України**

**Національний авіаційний університет**

**НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ КОМП’ЮТЕРНИХ**

**ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**

**Кафедра прикладної інформатики**

**Конспект лекцій**

з дисципліни «**Теорія та технології проектування**»

за спеціальністю 122 «Комп'ютерні науки та Інформаційні технології» (шифр та повна назва напряму (спеціальності))

Укладач: к.т.н. Ходаков Д,В.

(науковий ступінь, вчене звання, П.І.Б. викладача)

Конспект лекцій розглянутий та схвалений

на засіданні кафедри прикладної

 інформатики

Протокол № \_\_\_\_ від «\_\_\_»\_\_\_\_\_20\_\_р.

Завідувач кафедри\_\_\_\_\_\_\_\_Гамаюн В. П.

**Тема 2.1.1. Основні поняття та методологія проектування складних об’єктів і систем.**

Життєвий цикл інформаційної системи. Визначення структури інформаційної системи. Послідовність проектування складних систем. Технологічна мережа проектування.

Життєвий цикл інформаційної системи.

Методологія проектування інформаційних систем описує процес створення і супроводу систем у вигляді життєвого циклу (ЖЦ) ІС, представляючи його як деяку послідовність стадій і виконуваних на них процесів. Для кожного етапу визначаються склад і послідовність виконуваних робіт, одержувані результати, методи і засоби, необхідні для виконання робіт, ролі і відповідальність учасників і т.д. Таке формальне опис ЖЦ ІС дозволяє спланувати та організувати процес колективної розробки і забезпечити управління цим процесом.

Життєвий цикл ІС можна уявити як ряд подій, що відбуваються з системою в процесі її створення і використання.

Модель життєвого циклу відображає різні стани системи, починаючи з моменту виникнення необхідності в даній ІС і закінчуючи моментом її повного виходу з ужитку. Модель життєвого циклу - структура, яка містить процеси, дії і завдання, які здійснюються в ході розробки, функціонування та супроводження програмного продукту протягом усього життя системи, від визначення вимог до завершення її використання.

В даний час відомі і використовуються наступні моделі життєвого циклу:

Каскадна модель (рис. 2.1) передбачає послідовне виконання всіх етапів проекту в строго фіксованому порядку. Перехід на наступний етап означає повне завершення робіт на попередньому етапі.

Поетапна модель з проміжним контролем (рис. 2.2). Розробка ІС ведеться ітераціями з циклами зворотного зв'язку між етапами. Міжетапні коригування дозволяють враховувати реально існуюче взаємовплив результатів розробки на різних етапах; час життя кожного з етапів розтягується на весь період розробки.

Спіральна модель (рис. 2.3). На кожному витку спіралі виконується створення чергової версії продукту, уточнюються вимоги проекту, визначається його якість і плануються роботи наступного вітка.Особое увага приділяється початкових етапах розробки - аналізу і проектування, де реалізація тих чи інших технічних рішень перевіряється і обгрунтовується за допомогою створення прототипів (макетування) .



Мал. 2.1. Каскадна модель ЖЦ ІС 

Мал. 2.2. Поетапна модель з проміжним контролем



Мал. 2.3. Спіральна модель ЖЦ ІС

На практиці найбільшого поширення набули дві основні моделі життєвого циклу:

каскадна модель (характерна для періоду 1970-1985 рр.);

спіральна модель (характерна для періоду після 1986.г.).

У ранніх проектах досить простих ІС кожен додаток являло собою єдиний, функціонально та інформаційно незалежний блок. Для розробки такого типу додатків ефективним виявився каскадний спосіб. Кожен етап завершувався після повного виконання та документального оформлення всіх передбачених робіт.

Можна виділити наступні позитивні сторони застосування каскадного підходу:

на кожному етапі формується закінчений набір проектної документації, який відповідає критеріям повноти і узгодженості;

виконувані в логічній послідовності етапи робіт дозволяють планувати терміни завершення всіх робіт і відповідні витрати.

Каскадний підхід добре зарекомендував себе при побудові відносно простих ІС, коли на самому початку розробки можна досить точно і повно сформулювати всі вимоги до системи. Основним недоліком цього підходу є те, що реальний процес створення системи ніколи повністю не вкладається в таку жорстку схему, постійно виникає потреба в поверненні до попередніх етапах і уточнення або перегляд раніше прийнятих рішень. В результаті реальний процес створення ІС виявляється відповідним поетапної моделі з проміжним контролем.

Однак і ця схема не дозволяє оперативно враховувати виникаючі зміни і уточнення вимог до системи. Узгодження результатів розробки з користувачами проводиться тільки в точках, планованих після завершення кожного етапу робіт, а загальні вимоги до ІС зафіксовані у вигляді технічного завдання на весь час її створення. Таким чином, користувачі часто отримують систему, не задовольняє їх реальним потребам.

Спіральна модель ЖЦ була запропонована для подолання перерахованих проблем. На етапах аналізу і проектування реалізація технічних рішень і ступінь задоволення потреб замовника перевіряється шляхом створення прототипів. Кожен виток спіралі відповідає створенню працездатного фрагмента або версії системи. Це дозволяє уточнити вимоги, цілі та характеристики проекту, визначити якість розробки, спланувати роботи наступного витка спіралі. Таким чином поглиблюються і послідовно конкретизуються деталі проекту і в результаті вибирається обгрунтований варіант, який задовольняє дійсним вимогам замовника і доводиться до реалізації.

Ітеративна розробка відображає об'єктивно існуючий спіральний цикл створення складних систем. Вона дозволяє переходити на наступний етап, не чекаючи повного завершення роботи на поточному та вирішити головну задачу - якомога швидше показати користувачам системи працездатний продукт, тим самим активізуючи процес уточнення і доповнення вимог.

Основна проблема спірального циклу - визначення моменту переходу на наступний етап. Для її рішення вводяться тимчасові обмеження на кожен з етапів життєвого циклу, і перехід здійснюється відповідно до плану, навіть якщо не вся запланована робота закінчена. Планування проводиться на основі статистичних даних, отриманих в попередніх проектах, і особистого досвіду розробників.

Незважаючи на наполегливі рекомендації компаній - вендорів і експертів в області проектування і розробки ІС, багато компаній продовжують використовувати каскадну модель замість будь-якого варіанту итерационной моделі. Основні причини, за якими каскадна модель зберігає свою популярність, такі [2.1]:

1.Звичка - багато ІТ-фахівці здобували освіту в той час, коли вивчалася тільки каскадна модель, тому вона використовується ними і в наші дні.

2. Ілюзія зниження ризиків учасників проекту (замовника і виконавця). Каскадна модель припускає розробку закінчених продуктів на кожному етапі: технічного завдання, технічного проекту, програмного продукту і призначеної для користувача документації. Розроблена документація дозволяє не тільки визначити вимоги до продукту наступного етапу, а й визначити обов'язки сторін, обсяг робіт і терміни, при цьому остаточна оцінка термінів і вартості проекту проводиться на початкових етапах, після завершення обстеження. Очевидно, що якщо вимоги до інформаційної системи змінюються в ході реалізації проекту, а якість документів виявляється невисоким (вимоги неповні та / або суперечливі), то в дійсності використання каскадної моделі створює лише ілюзію визначеності і на ділі збільшує ризики, зменшуючи лише відповідальність учасників проекту. При формальному підході менеджер проекту реалізує тільки ті вимоги, які містяться в специфікації, спирається на документ, а не на реальні потреби бізнесу. Є два основних типи контрактів на розробку ПЗ. Перший тип передбачає виконання певного обсягу робіт за певну суму у визначений термін (fixed price). Другий тип передбачає погодинну оплату роботи (time work). Вибір того чи іншого типу контракту залежить від ступеня визначеності завдання. Каскадна модель з певними етапами і їх результатами краще пристосована для укладення контракту з оплатою за результатами роботи, а саме цей тип контрактів дозволяє отримати повну оцінку вартості проекту до його завершення. Більш імовірно укладання контракту з погодинною оплатою на невелику систему, з відносно невеликою вагою в структурі витрат підприємства. Розробка і впровадження інтегрованої інформаційної системи вимагає істотних фінансових витрат, тому використовуються контракти з фіксованою ціною, і, отже, каскадна модель розробки і впровадження. Спіральна модель частіше застосовується при розробці інформаційної системи силами власного відділу ІТ підприємства.

3. Проблеми впровадження при використанні ітераційної моделі. У деяких областях спіральна модель не може застосовуватися, оскільки неможливо використання / тестування продукту, що володіє неповною функціональністю (наприклад, військові розробки, атомна енергетика і т.д.). Поетапне ітераційне впровадження інформаційної системи для бізнесу можливо, але пов'язане з організаційними труднощами (перенесення даних, інтеграція систем, зміна бізнес-процесів, облікової політики, навчання користувачів). Трудовитрати при поетапному ітераційне впровадженні виявляються значно вище, а управління проектом вимагає справжнього мистецтва. Передбачаючи вказані складнощі, замовники вибирають каскадну модель, щоб "впроваджувати систему один раз".

Кожна зі стадій створення системи передбачає виконання певного обсягу робіт, які подаються у вигляді процесів ЖЦ. Процес визначається як сукупність взаємопов'язаних дій, що перетворюють вхідні дані у вихідні. Опис кожного процесу включає в себе перелік вирішуваних завдань, вихідних даних і результатів.

Існує цілий ряд стандартів, що регламентують ЖЦ ПЗ, а в деяких випадках і процеси розробки.

Значний внесок у теорію проектування та розробки інформаційних систем внесла компанія IBM, запропонувавши ще в середині 1970-х років методологію BSP (Business System Planning - методологія організаційного планування). Метод структурування інформації з використанням матриць перетину бізнес-процесів, функціональних підрозділів, функцій систем обробки даних (інформаційних систем), інформаційних об'єктів, документів і баз даних, запропонований в BSP, використовується сьогодні не тільки в ІТ-проектах, але і проектах по реінжинірингу бізнес -процесів, зміни організаційної структури. Найважливіші кроки процесу BSP, їх послідовність (отримати підтримку вищого керівництва, визначити процеси підприємства, визначити класи даних, провести інтерв'ю, обробити і організувати дані інтерв'ю) можна зустріти практично у всіх формальних методиках, а також в проектах, що реалізуються на практиці.

Серед найбільш відомих стандартів можна виділити наступні:

ГОСТ 34.601-90 - поширюється на автоматизовані системи та встановлює стадії і етапи їх створення. Крім того, в стандарті міститься опис змісту робіт на кожному етапі. Стадії і етапи роботи, закріплені в стандарті, більшою мірою відповідають каскадної моделі життєвого циклу [2.2].

* ISO / IEC 12207: 1995 - стандарт на процеси і організацію життєвого циклу. Поширюється на всі види замовного ПЗ. Стандарт не містить опису фаз, стадій і етапів [2.3].
* Custom Development Method (методика Oracle) з розробки прикладних інформаційних систем - технологічний матеріал, деталізований до рівня заготовок проектних документів, розрахованих на використання в проектах із застосуванням Oracle. Застосовується CDM для класичної моделі ЖЦ (передбачені всі роботи / завдання та етапи), а також для технологій "швидкої розробки" (Fast Track) або "полегшеного підходу", рекомендованих в разі малих проектів.
* Rational Unified Process (RUP) пропонує ітеративну модель розробки, що включає чотири фази: початок, дослідження, побудова та впровадження. Кожна фаза може бути розбита на етапи (ітерації), в результаті яких випускається версія для внутрішнього або зовнішнього використання. Проходження через чотири основні фази називається циклом розробки, кожен цикл завершується генерацією версії системи. Якщо після цього робота над проектом не припиняється, то отриманий продукт продовжує розвиватися і знову мине ті ж фази. Суть роботи в рамках RUP - це створення і супровід моделей на базі UML [2.4].
* Microsoft Solution Framework (MSF) схожа з RUP, так само включає чотири фази: аналіз, проектування, розробка, стабілізація, є ітераційної, припускає використання об'єктно-орієнтованого моделювання. MSF в порівнянні з RUP більшою мірою орієнтована на розробку бізнес-додатків.
* Extreme Programming (XP). Екстремальне програмування (найновіша серед розглянутих методологій) сформувалося в 1996 році. В основі методології командна робота, ефективна комунікація між замовником і виконавцем протягом усього проекту з розробки ІС, а розробка ведеться з використанням послідовно допрацьовувати прототипів.
* Відповідно до базовим міжнародним стандартом ISO / IEC 12207 всі процеси ЖЦ ПО діляться на три групи:

1.Основні процеси:

* придбання;
* поставка;
* розробка;
* експлуатація;
* супровід.

2.Допоміжні процеси:

* документування;
* управління конфігурацією;
* забезпечення якості;
* розв'язати проблеми;
* аудит;
* атестація;
* спільна оцінка;
* верифікація

3.Організаційні процеси:

* створення інфраструктури;
* управління;
* навчання;
* удосконалення.

Визначення структури інформаційної системи.

Структуру інформаційної сістемисоставляет сукупність окремих її частин, які називаються підсистемами. Підсистема - це частина системи, виділена за будь-якою ознакою. Загальну структуру інформаційної системи можна розглядати як сукупність підсистем незалежно від сфери застосування.

Послідовність проектування складних систем.

Розглянемо основні етапи проектування з позицій технології обробки інформації.

Традиційно проектування складних технічних систем підрозділяють на наступні етапи або стадії розробки (рис. 3.2):

· Технічне завдання на проектований об'єкт;

· науково-дослідна робота ;

· ескізний проект;

· Технічний проект;

· робочий проект;

· Технологія виготовлення і випробування спроектованого об'єкта (дослідного зразка або партії), внесення корекції (при необхідності).



Мал. 3.2.Етапи проектування складних систем

Технічне завдання

На етапі розробки технічного завдання (ТЗ) вирішуються такі завдання:

· Пошук і вибір необхідної науково-технічної інформації (про прототипи, патентних даних і т.д.) з відповідної бази даних. Нова схема (пристрій) може або мати, або не мати аналогів. У разі, якщо аналоги є, можна приступити до етапу проектування пристрою (системи). Але, як правило, аналога немає або що розробляється система повинна перевершувати відомий аналог, тому необхідне проведення етапу НДР;

· Аналіз обраної інформації та формулювання на його основі технічних вимог (ТТ) до проектованого об'єкту. Оформлення ТТ відповідно до встановлених правил.

На даному етапі проектування можуть бути автоматизовані операції пошуку інформації та оформлення документів. Може бути також автоматизована деяка частина допоміжних дій з аналізу обраної інформації, наприклад, угруповання її за заданими ознаками, вибір найменш або найбільш порівнянних між собою варіантів і т. Д.

Крім того, на етапі ТЗ вирішуються і оформляються у відповідних документах, наприклад, такі питання:

· Перерахування функцій, які виконуються пристроєм;

· Розробка структурної схеми пристрою;

· Оформлення умов працездатності пристрою;

· Оформлення вимог до вихідних параметрів;

· Визначення характеристик окремих вузлів;

· Розробка алгоритмів виконуваних операцій.

етап НДР

Цей етап є попередніми проектуванням. Це один з найвідповідальніших етапів. Для вирішення завдань цього етапу необхідно використання ЕОМ. Це так звані автоматизовані системи наукових досліджень (АСНИ).

На етапі НДР необхідне рішення наступних завдань:

1. Формулювання критеріїв якості та управління.

2. Управління науковим експериментом.

3. Проведення пасивного або (і) активного експерименту з обробкою їх результатів.

4. Розробка математичних моделей і їх ідентифікація за експериментальними даними.

5. Відпрацювання технологічних процесів виготовлення об'єктів РЕМ з метою пошуку норм на параметри, що забезпечують оптимальні вихідні показники якості.

6. Формування узагальненого критерію якості, що включає в себе всі приватні показники якості. Узагальнений критерій приймається далі за цільову функцію при вирішенні задачі оптимізації.

7. Рішення завдання оптимізації. Проводиться варіювання вхідних і керуючих параметрів технологічного процесу в рамках встановлених норм (допусків) з метою отримання оптимального критерію якості.

8. Пошук принципову можливість побудови системи.

9. Розробка нових технічних засобів, в тому числі засобів контролю та вимірювань.

В результаті проведення НДР видається Технічне Пропозиція (ТП).

Хоча етап НДР є самостійним етапом, тут можуть використовуватися методи, алгоритми та програми з САПР.

Це етап ескізного проектування. На даному етапі проводиться наступне рішення задач:

1. Розробляється ескіз проектованої системи (пристрої) з детальною розробкою її можливостей, здійснюється пошук і вибір більш детальної інформації.

2. На основі аналізу отриманої інформації приймають попередні проектні рішення і оформляють перші проектні документи.

3. Для вироблення проектних документів виробляють різні розрахунки, зміст, обсяг і трудомісткість яких залежать від характеристик об'єкта, що проектується.

Роботи цього етапу в найбільшою мірою піддаються автоматизації, і їх автоматизація дає найбільший техніко-економічний ефект за рахунок оптимізації проектних рішень.

Автоматизація зазначених робіт досягається за рахунок застосування оптимізаційних математичних методів.

Етап розробки технічного проекту об'єкту

На цьому етапі деталізують і уточнюють рішення, прийняті при ескізному проектуванні, і створюють нові, більш точні проектні документи. Знову проводять пошук, вибір і аналіз вихідної інформації (в основному нормативно-технічної та техніко-економічної). Знову виконують численні розрахунки, але вже за іншими, більш точним методикам. Ці роботи в значній мірі можуть бути автоматизовані.

Більшість документів, сформованих на етапах ескізного і технічного проектування, використовуються тільки для виконання робочого проектування і не входять до складу робочої та експлуатаційної документації. Інформація, напрацьована на розглянутих стадіях, служить вихідною для робочого проектування. Це означає, що в умовах автоматизованого проектування доцільно створити банки тимчасової інформації по об'єкту, що проектується.

**Технологічна мережа проектування**

Реальний процес проектування ІС відображається в технологічній мережі проектування.

Технологічна мережа проектування (ТМП) — взаємопов'язана за входом і виходом послідовність ТО проектування, виконання яких має забезпечити створення проекту ІС.

Складанню ТМП передує ознайомлення з економічним ОУ, що дає змогу одержати загальне уявлення про нього, сформулювати основні цілі та задачі проектування, визначити перелік основних комплексів робіт, у тому числі ТО.

Проектування ІС — складний процес роботи багатьох виконавців, що включає багато різноманітних робіт і потребує суворої впорядкованості, певної послідовності та планомірності їх виконання.

Найпоширенішим методом планування й управління розробленням і впровадженням проекту є система планування мереж та управління — СПУ (РЕRТ), за допомогою якої можна отримати уявлення про всю ТМП, що забезпечує найраціональнішу послідовність проектних робіт.

Порядок виконання робіт зі створення ІС подається у вигляді графа мережі, який включає детальний опис проектування і містить багато операцій (робіт).

Збільшений граф мережі за стадіями й етапами проектування дає змогу простежити розвиток системи від початку робіт з її створення до введення в експлуатацію. Його параметри обчислюють на ЕОМ із застосуванням спеціального ППП.

Застосування графіків мереж для організації управління процесом проектування ІС дає змогу визначити його загальну трудомісткість і на основі існуючих нормативів з'ясувати потрібну кількість ресурсів на виконання проектних робіт. Календаризація графіка мережі здійснюється на основі наявності трудових і технічних ресурсів. Для цього розробляють календарний графік для виконавців з урахуванням застосовуваних ними технічних засобів у процесі проектних робіт.

Визначення критичного шляху і резервів за ТО дає змогу контролювати виконання проектних робіт, оперативно керувати процесом проектування, перерозподіляючи роботи між виконавцями-розробниками системи або навпаки (виконавців між роботами) з метою створення проекту ІС у суворо встановлені терміни.

Використання сучасних засобів автоматизації проектування ІС допускає оптимізацію графіка мережі, що значно скорочує терміни створення проекту, знижує витрати праці на його розроблення.

Отже, СПУ для організації процесу проектування ІС вказує на те, що під час роботи з ТМП може бути використаний математичний апарат.

Якщо в ТМП у жодній ТО не застосовуються засоби проектування, тобто всі подані в ній операції виконуються вручну, то таку мережу називають канонічною ТМП, яка відображає процес створення ІС для конкретного ОУ. У ній цей процес подається на найнижчому рівні декомпозиції і є базою для обґрунтування застосування та розроблення різних засобів проектування (ТПР, ППП тощо).

Якщо до канонічної ТМП застосувати метод композиції, виділяючи в ній певні фрагменти, то для кожного з них можна визначити спільні вхід, вихід і синтезувати перетворювач, тобто збільшити ТО. За допомогою таких ТО створюють засоби, які дають змогу автоматизувати виконання проектних робіт у межах виділених фрагментів. Відбувається збільшення ТМП, вона стає коротшою, виникає можливість утворення САПР ІС, що автоматизує весь процес її створення і є узагальненим перетворювачем, на вході якого подано матеріали обстеження об'єкта, вимоги до нової системи управління та ресурси проектування, а на виході формується потрібний проект ІС.

Використовуючи канонічну мережу проектування, можна побудувати ТМП, орієнтовані на певні категорії спеціалістів, керівників, аналітиків, програмістів та ін. У таких мережах докладно описуються ділянки, які їх цікавлять, а інші призначаються для збільшення.

Канонічна мережа проектування може бути основою для порівняння двох і більше альтернативних ТМП. Наприклад, існують дві ТМП ІС для ОУ. З метою вибору кращої порівнюють ресурси (вартісні, трудові) та їх проектування й еквівалентність за здобутим результатом.

Отже, процес проектування ІС можна формально описати за допомогою ТМП. Якщо відомо повний набір ТО, потрібних для створення відповідного проекту, то існує також формалізований алгоритм побудови ТМП.

Серед вихідної множини ТО Т = {Тi} є альтернативні сукупності їх, що ведуть до створення сукупності альтернативних ТМП, з якої слід вибрати конкретну, що відповідала б умовам вартості, трудомісткості та ін.

Приведений нижче сценарій демонструє використання систем CAD/CAM/ САЕ в рамках усього життєвого циклу продукту для досягнення згаданих цілей: підвищення якості (Q), зниження вартості (С) и прискорення відвантаження (D). Цей сценарій може показатися дещо спрощеним на фоні сучасних передових комп’ютерних технологій, однак він ілюструє напрям розвитку техніки. Розглянемо фази розробки и виробництва шафи для аудіосистеми. Життєвий цикл цього продукту буде схожим на життєвий цикл механічної системи чи споруди (будівлі), а значить, наш сценарій буде застосованим і до таких продуктів.

Нехай в технічних вимогах для розробника вказано, що шафа повинна мати чотири полички: одну для програвача компакт-дисків, одну для програвача аудіокасет, одну для радіоприймача и одну для збереження компакт-дисків. Ймовірно, розробник зробить більшість ескізів конструкції, перед тим як прийде до кінцевого варіанту. На даному етапі він може користуватися ароматизованою системою розробки робочих креслень (якщо задача розв’язується в двох вимірах) або системою геометричного моделювання (у випадку трьох вимірів). Концептуальний проект може бути відправлений в відділ маркетингу по електронній почті для отримання відгуку. Взаємодія розробника з відділом маркетингу може проходити і в реальному часі через об’єднання в мережу комп’ютери. При наявності підходящого обладнання подібна взаємодія може бути зручною і продуктивною. Інформація про готовий концептуальний проект зберігається в базі даних. Туди попадають відомості про конфігурації меблів (в нашому випадку — вертикальне зберігання компонентів аудіосистеми один над одним), кількості полиць, розподіленні полиць по компонентам і тому подобні дані. Іншими словами, всі особливості проекту впорядковуються і поміщаються в базу даних з можливістю зчитування і змінювання в будь-який наступний момент.

Наступний крок — визначення розмірів шафи. Його габарити повинні бути вибрані таким чином, щоб на кожну поличку можна було поставити одну з множини, які є ринку моделей аудіотехніки відповідного класу. Значить, потрібно отримати відомості про їх розміри. Ці відомості можна взяти в каталозі чи в базі даних виробників чи постачальників. Доступ до бази даних здійснюється аналогічно доступу до книг і їх змісту при підключені до електронної бібліотеки. Розробник може навіть скопіювати відомості в свою власну базу даних, якщо він планує часто користуватися ними. Накопичення відомостей про проект подібно накопиченню форм і файлів при роботі з текстовими процесорами. Форма конструкції повинна змінюватися у відповідності з отриманими відомостями. Потім розробник повинен вибрати матеріал для шафи. Він може взяти натуральний дуб, сосну, ДСП, або ще щось інше. В нашому випадку вибір здійснюється інтуїтивно або виходячи з досвіду розробника. Хоча у випадку продуктів, які розраховані на роботу в жорстких умовах, наприклад, механічних приладів, розробник обов’язково враховує властивості матеріалів. На цьому етапі також необхідна база даних, тому що в ній можуть бути збережені властивості багатьох матеріалів. Можна скористуватися навіть експертною системою, яка вибере матеріал по властивостям, що зберігаються в базі даних. Інформація про вибраний матеріал також розміщується в базі. Наступний крок — визначення товщини полок, дверець і бічних стінок. В простому випадку, який розглядається, товщина може визначатися головним чином естетичними поглядами. Хоча вона повинна бути по крайній мірі достатньою для того, щоб уникнути прогину під дією встановленої в шафі техніки. В механічних приладах високою точністю і структурах, які розраховані на великі навантаження, такі параметри, як товщина, повинні визначатися точним розрахунком, щоб уникнути деформації. Для розрахунку деформації структур широко використовується метод кінцевих елементів. Метод кінцевих елементів застосовується до аналітичної моделі конструкції. В нашому випадку аналітична модель складається з каркасних сіток, на які розбивається шафа. Перехід до наближення листів може бути виконано автоматично за допомогою алгоритму перетворення до середніх вісей (medial axis transformation — MAT). Елементи оболонки наближення листів також можуть генеруватися автоматично.

Параметри навантаження, які в цьому випадку є просто вага відповідних пристроїв, які вираховуються з бази даних так само, як і відомості про розміри. Визначаючи залежність прогину полок від їх товщини, розробник може вибрати необхідне значення його параметра і зберегти його в базі даних. Цей процес може бути автоматизований шляхом інтеграції методу кінцевих елементів з процедурою оптимізації. Аналогічним чином можна визначити товщину бічних стінок і дверцят, однак зробити це можна і просто з естетичних міркувань.

Потім розробник вибирає метод збірки полок і бічних стінок. В ідеалі метод також може бути визначений з розрахунку міцності структури в цілому або за допомогою експертної системи, що має відомості про методи збірки.

Після завершення етапів концептуалізації проекту, його аналізу і оптимізації розробник переходить до роботи над проектною документацією, яка описує шафу з точністю до маленьких дрібниць. Креслення окремих деталей (полок, дверцят і бічних стінок) виготовлюються в системі розробки робочих креслень. На цьому етапі розробник може додавати деякі естетичні деталі, наприклад декоративні елементи на дверцятах і бічних стінках. Детальні креслення розміщують в базі даних для використання в процесі виробництва. Виготовлення шафи здійснюється в наступному порядку. Форма кожної деталі наноситься на необробленому матеріалі (в нашому випадку дерево) і вирізається пилою. Кількість відходів можна знизити, розміщуючи деталі на кусках дерева оптимально. Розробник може перевіряти різні варіанти розміщення на екрані комп’ютера до тих пір, поки не буде знайдена конфігурація з мінімальною кількістю відходів. Комп’ютерна програма може допомогти в цій роботі, розраховуючи кількість відходів для кожної конфігурації. Програма більш високого рівня може самостійно визначити найбільш економічне розміщення деталей на заготовці. В будь-якому випадку кінцева конфігурація зберігається в комп’ютері і використовується для розрахунку траєкторії руху пили станка з числовим програмним управлінням. Більш того, програмні засоби дозволяють розробити зажими і кріплення для процедури випилювання, а також запрограмувати системи передачі матеріалу. Ці системи можуть бути як простими конвеєрами, так і складними роботами, що передають необроблений матеріал на розпилку і забираючи готові деталі. Підготовлені деталі повинні бути складені разом. Процес збірки також може виконуватися роботами, які програмуються автоматично на основі опису кінцевого продукту і його деталей, що зберігаються в базі даних. Одночасно проектуються зажими і кріплення для автоматизованої збірки. Потім, робот може бути запрограмований на окраску шафи після збірки. Зажими і кріплення для збірки проектуються або вибираються планувальником процесів, а програмування роботів здійснюються в інтерактивному режимі шляхом переміщення робочого органа робота вручну.

Загальний вигляд отриманого сценарія показаний на рис. 1, з якого видно, яким чином база даних дозволяє інтегрувати системи CAD, CAE і САМ, що і являється кінцевою метою CIM.

**Тема 1.2. Системний (структурний) рівень комп’ютерного проектування складних об’єктів та систем.**

Програмні та апаратні складові складних об’єктів та систем. Основні вимоги. Проектування спеціалізованих комп’ютерних систем на системному рівні. Базові структури

**ОСОБЛИВОСТІ ПРОЕКТУВАННЯ СКС**

Спеціалізовані комп’ютерні системи вирізняється великою різноманітністю архітектур та методик проектування. Основними чинниками, що визначають проектування СКС є:

* висока складність;
* поєднання аналогової та цифрових складових;
* поєднання як апаратно орієнтованих компонент з програмованими процесорами
* високі вимоги до засобів обміну інформації між компонентами системи;
* жорсткі вимоги до продуктивності, вартості, споживаної потужності і.т.д.;
* реалізація як СНК;
* обмеження на час проектування та підготовки до масового випуску;
* можливість модифікації системи.



Наявність вищенаведених чинників та постійне зростання складності СКС призводить до постійного розриву між можливостями сучасної елементної бази та продуктивністю розробників.



З метою зменшення наведеного розриву, розробка системи повинна виконуватись з використанням засобів системного проектування, сучасних засобів розробки програмних та апаратних компонент. Використання мови асемлер та проектування на вентильному рівні суттєво збільшує час проектування та є недопустимим. Необхідним є коректне розбиття задачі на програмну та апаратні компоненти вже на системному рівні та визначення об’єму робіт над даними компонентами. Перед початком робіт над розробкою програмних та апаратних складових, необхідно чітко сформулювати основних їх вимоги.

При цьому, вимоги що ставляться до програмної складової є:

* гнучкість;
* реконфігурованість, можливість модифікації;
* складність функціонування;
* вартість.

До апаратної складової:

* висока швидкодія;
* низьке споживання;
* низька вартість при великих тиражах.

При проектуванні СКС прийнято виділяти наступні рівні проектування, які є загальними для всіх послідовностей проектування та визначають рівень опису системи.

* Системний рівень —на даному рівні СКС представляється та описується в формі конкурентних процесів, кожен з яких виконує певні функції. Результатом даного етапу є отримання структурної схеми СКС у вигляді процесорів, спеціалізованих пристроїв та шин. Формуються чіткі вимоги щодо функціонування кожного елемента СКС, формуються часткові ТЗ.
* Високий-рівень – на даному рівні СКС представляється у вигляді поведінкового опису. Програмні складові системи представляються описом роботи алгоритмів мовою програмування. Апаратні складові кодуються мовою опису апаратних засобів на рівні міжрегістрових передач. Результатом даного рівня є отримання двійкового коду програм та функціональної схами пристрою.
* Логічний рівень – компоненти системи представляються на рівні логічних схем, - виконується синтез комбінаційних та послідовнісних схем на рівні примітивних логічних елементів. Результатом даного рівня є отримання принципової схеми (netlist) пристрою.
* Фізичний рівень —на основі принципової схеми виконується виготовлення СКС.

Послідовність проектування СКС може буде представлена наступною спрощеною послідовністю проектування.



*Специфікація системи (System Specification).* Виконується опис та моделювання роботи системи на рівні функціонування основних вузлів та блоків. Для специфікація системи та її опису використовуються засоби та мови програмування/проектування такі як:

* Мови специфікації такі як UML, SDL.
* Мови програмування C, C++, Java, Esterel, assembly languages.
* Мови опису апаратних засобів VHDL, Verilog, SystemC.

*Проектування*

Оцінка Проводиться оцінка затрат обладнання на реалізацію апаратних компонент, об’єм програмного коду, оцінюється продуктивність. Актуальною є задачи оптимального розбиття на апаратні та програмні складові системи. Вибирається базова структура СКС та послідовність проектування.

*Розбиття та проектування складових* Програми компілюються у асемблерні коди мікропроцесора, можуть бути використані ОС реального часу, апаратні компонети представляється мовами опису апаратних засобів. Проводиться розробка шин та протоколів обміну інформації між компонентами системи.

Як зазначалось у попередніх підрозділах, розроблення та використання СППР та ЕС безпосередньо пов’язані з моделюванням у тих проблемних галузях, для яких створюються відповідні інформаційні системи. Крім того, моделювання є основним етапом системного аналізу. Тому, звичайно, існують сучасні інформаційні технології, які забезпечують автоматизацію цього процесу. Традиційно під моделюванням на ЕОМ розумілося лише імітаційне моделювання. Але в останні роки завдяки розвитку графічного інтерфейсу і графічних пакетів широкий розвиток отримало комп’ютерне, структурно-функціональне моделювання та було покладено початок використанню комп’ютера при концептуальному моделюванні, де він використовується, наприклад, для побудови систем штучного інтелекту. Отже, поняття «комп’ютерне моделювання» значно ширше за традиційне поняття «моделювання на ЕОМ». Підкомп’ютерною моделлю найчастіше розуміють: умовний образ об’єкта чи деякої системи об’єктів (або процесів), описаних за допомогою взаємозалежних комп’ютерних таблиць, блок-схем, діаграм, графіків, малюнків, анімаційних фрагментів, гіпертекстів тощо, які відображують структуру та взаємозв’язки між елементами об’єкта чи системи. Комп’ютерні моделі такого типу називають структурно-функціональними; окрему програму, сукупність програм, програмний комплекс, що дає змогу через певну послідовність обчислень та графічне відображення їх результатів, відтворювати (імітувати) процеси функціонування об’єкта чи системи об’єктів за умови впливу на них різних, як правило випадкових, факторів. Такі моделі називають імітаційними моделями. Комп’ютерне моделювання — це метод розв’язання завдання аналізу або синтезу складної системи на засадах використання його комп’ютерної моделі. Суть комп’ютерного моделювання полягає в знаходженні кількісних і якісних результатів за допомогою наявної моделі. Якісні висновки, які отримують за результатами аналізу, дають змогу знайти невідомі раніше характеристики складної системи: її структуру, динаміку розвитку, стійкість, цілісність тощо. Кількісні висновки, головно, мають характер прогнозу майбутніх чи пояснення минулих значень змінних, що характеризують систему. Предметом комп’ютерного моделювання можуть бути: економічна діяльність фірми, банку, виробничого підприємства; інформаційно-обчислювальна мережа; технологічний процес; будь-який інший реальний об’єкт чи процес, наприклад процес інфляції, і загалом, будь-яка складна система. Цілі комп’ютерного моделювання можуть бути різними, однак найчастіше моделювання є, як уже зазначалося раніше, головним етапом (процедурою) системного аналізу, тобто сукупності методологічних засобів, що використовуються для підготовки та прийняття рішень економічного, організаційного, соціального чи технічного характеру. Комп’ютерна модель складної системи має за можливості відображати всі головні фактори і взаємозв’язки, що характеризують реальні ситуації, критерії та обмеження. Модель має бути досить універсальною, щоб за можливості була спроможною описувати близькі за призначенням об’єкти, і водночас досить простою, щоб уможливлювати виконання необхідних досліджень з мінімальними витратами. Усе це підтверджує той факт, що моделювання систем, розглянуте загалом, є скоріш мистецтвом, ніж наукою, із самостійним набором засобів відображення явищ і процесів реального світу. Тому досить складно навести єдину, узагальнену класифікацію завдань комп’ютерного моделювання та створити достатньо універсальні його інструментальні засоби для об’єктів довільної природи. Однак, якщо звузити коло розглянутих об’єктів, обмеживши його, наприклад, завданнями комп’ютерного моделювання за системного аналізу об’єктів економіко-організаційного керування, то можна підібрати ряд досить універсальних підходів і програмних засобів. Існує велике різноманіття засобів комп’ютерного моделювання, особливо структурно-функціонального, які з’являються мало не щодня. Однією із спроб подолати розбіжності між такими засобами є уніфікована мова моделювання. Уніфікована мова моделювання (Unified modeling language, UML) є графічною мовою для візуалізації, специфікації, конструювання та документування систем, в яких більша роль належить програмному забезпеченню. За допомогою UML можна розробити детальний план системи, що відображує не тільки її концептуальні елементи, такі як системні функції та бізнес-процеси, а й конкретні особливості реалізації, в тому числі типи, написані спеціальними мовами програмування. Можна розробити також схеми баз даних та програмні компоненти багаторазового використання. 1996 року група управління об’єктами (Object Management Group, OMG) звернулась до об’єктно орієнтованої спільноти з пропозицією створити стандартний синтаксис для об’єктно орієнтованого аналізу та відповідну семантичну метамодель. Перша версія UML (UML 1.0) з’явилась у січні 1997 р. як відповідь на цю пропозицію. Після її обговорення та дороблення в листопаді 1997 р. версія UML 1.1 була успішно затверджена та прийнята до використання практично всіма найбільшими компаніями — виробниками програмного забезпечення (Microsoft, IBM, Hewlett-Packard, Oracle, Sybase та ін.). Крім того, практично всі світові виробники CASE-засобів, крім Rational Software, заявили про готовність підтримки UML у своїх продуктах. Творці UML подають її як мову для визначення, подання, проектування та документування програмних систем, бізнес-систем та інших різних систем. UML виключає нотацію та метамодель. Нотація є сукупністю графічних об’єктів, які використовуються в моделях, вона є синтаксисом мови моделювання. Для детальнішого ознайомлення з ключовими поняттями, семантикою та процесом використання UML можна порекомендувати [43]. Основні принципи та концептуальні засади CASE-технологій Тенденції сучасних інформаційних технологій ведуть до постійного ускладнення інформаційних систем (ІС), що створюються в різних галузях економіки. Сучасні великі проекти ІС мають, як правило, такі особливості: — складність описання (досить велика кількість функцій, процесів, елементів даних та складні взаємозв’язки між ними), що потребує ретельного моделювання, аналізу даних та процесів; — наявність сукупностей компонентів (підсистем), що тісно взаємодіють та мають свої локальні задачі і цілі функціонування (наприклад, традиційних додатків, пов’язаних з обробленням трансакцій та розв’язанням регламентних задач, та додатків аналітичного оброблення (підтримки прийняття рішень), що використовують нерегламентовані запити до даних великого об’єму); — відсутність прямих аналогів, що обмежує використання якихось типових проектних рішень та прикладних систем; — необхідність інтеграції додатків, що існують та тільки розробляються; — функціонування в неоднорідному середовищі на різних апаратних платформах; — розрізненість та різнорідність окремих груп розробників за рівнем кваліфікації та вкоріненими традиціями використання певних інструментальних засобів; — істотна тривалість проекту зумовлена, з одного боку, обмеженими можливостями колективу розробників та, з другого боку, масштабами організації замовника і різними рівнями готовності її підрозділів до впровадження ІС. Для успішного впровадження проекту об’єкт проектування (ІС) має бути, передусім, адекватно описаний, мають бути побудовані повні та несуперечливі функціональні та інформаційні моделі ІС. Накопичений досвід свідчить, що це логічно складна, трудомістка та тривала робота. Вона потребує високої кваліфікації спеціалістів, які беруть у ній участь. У 70-х та 80-х роках за розроблення ІС досить широко застосовували структурну методологію. Але її використання для проектування ІС викликало ряд проблем, зумовлених, зокрема, значним обсягом ручної роботи: — неадекватна специфікація вимог; — нездатність виявляти помилки в проектних рішеннях; — низька якість документації, що знижує експлуатаційні властивості; — затяжний цикл та незадовільні результати тестування. Перераховані вище проблеми спонукали до появи програмно-технологічних засобів спеціального типу — CASE-засобів, що реалізують CASE-технологію створення та супроводження ІС [11, 36]. Термін CASE (Computer Aided Software Engineering (комп’ютерна підтримка інженерії програмного забезпечення), а за іншою версією — Computer Aided System Engineering (комп’ютерна підтримка інженерії систем)) використовується зараз у досить широкому розумінні. Нині під терміном CASE-засоби розуміють програмні засоби, що підтримують процеси створення та супроводження ІС, враховуючи аналіз та формулювання вимог, проектування прикладного ПЗ (додатків) та баз даних, генерування коду, тестування, документування, забезпечення якості, конфігураційне керування та управління проектом, а також інші процеси. CASE-технологія є методологією проектування ІС, а також набором інструментальних засобів, що уможливлюють у наочній формі моделювання будь-якої проблемної галузі, аналіз цієї моделі на всіх етапах розроблення та супроводження ІС та розроблення додатків відповідно до інформаційних потреб користувачів. Більшість існуючих CASE-засобів ґрунтуються на методологіях структурного (головно) та об’єктно орієнтованого аналізу і проектування, що використовують специфікації у вигляді діаграм або текстів для описування зовнішніх вимог, зв’язків між моделями системи, динаміки поведінки системи та структури програмних засобів. Сучасні CASE-засоби охоплюють широкий діапазон підтримки численних технологій проектування: від простих засобів аналізу і документування до повномасштабних засобів автоматизації. До CASE-засобів належать як відносно дешеві системи для персональних комп’ютерів з дуже обмеженими можливостями, так і дорогі системи для неоднорідних обчислювальних платформ і операційних середовищ. Так, сучасний ринок програмних засобів нараховує близько 300 різних CASE-засобів, найпотужніші з яких певною мірою використовуються практично усіма провідними західними фірмами. До CASE-засобів здебільшого відносять будь-який програмний засіб, що використовується для автоматизації моделювання систем та має такі характерні риси: потужні графічні засоби для описування і документування, що забезпечують зручний інтерфейс із розробником і розвивають його творчі можливості; інтеграція окремих компонентів CASE-засобів, що забезпечує керованість процесом розроблення моделі; використання у спеціальний спосіб організованого сховища проектних метаданих (репозиторію). Інтегрований CASE-засіб (чи комплекс засобів) містить такі компоненти; репозиторій, що є основою CASE-засобу. Він повинен забезпечувати збереження версій проекту і його окремих компонентів, синхронізацію надходження інформації від різних розробників, контроль метаданих на повноту і несуперечність; графічні засоби аналізу і проектування, що забезпечують створення і редагування ієрархічно зв’язаних діаграм (DFD, ERD тощо); засоби розроблення додатків, включаючи мови 4GL і генератори кодів; засоби конфігураційного керування; засоби документування; засоби тестування; засоби керування проектом; засоби реінжинірингу. Усі сучасні CASE-засоби можуть бути класифіковані, головно, за типами і категоріями. Класифікація за типами відображує функціональну орієнтацію CASE-засобів на різні процеси моделі. Класифікація за категоріями визначає рівень інтегрованості за функціями, що можуть виконуватися, і включає окремі локальні засоби, які розв’язують невеликі автономні задачі (tools), набір частково інтегрованих засобів, що охоплюють більшість етапів моделювання системи (toolkit) і цілком інтегровані засоби, що підтримують весь цикл аналізу та проектування системи і зв’язані загальним репозиторієм. Крім цього, CASE-засоби можна класифікувати за такими ознаками: методологіями і моделями систем та БД, що застосовуються в CASE-засобах; рівнем інтегрованості із СУБД; доступними платформами. Класифікація за типами переважно збігається з компонентним складом CASE-засобів та включає такі основні типи: засоби аналізу (Upper CASE), призначені для побудови й аналізу моделей проблемної галузі (Design/IDEF (Meta Software), BPwin (Logic Works)); засоби аналізу і проектування (Middle CASE), що підтримують найрозповсюдженіші методології проектування і, які використовують для створення проектних специфікацій (Vantage Team Builder (Cayenne), Designer/2000 (ORACLE), Silverrun (CSA), PRO-IV (McDonnell Douglas), CASE-Аналітик (Макропроджект)). Виходом таких засобів є специфікації компонентів і інтерфейсів системи, структури системи, алгоритми і структури даних; засоби проектування баз даних, що забезпечують моделювання даних і генерування схем баз даних (як правило, мовою SQL) для найрозповсюдженіших СУБД. До них належать ERwin (Logic Works), S-Designor (SDP) і DataBase Designer (ORACLE). Засоби проектування баз даних наявні також у складі CASE-засобів Vantage Team Builder, Designer/2000, Silverrun і PRO-IV; засоби розробки додатків. До них належать засоби 4GL (Uniface (Compuware), JAM (JYACC), PowerBuilder (Sybase), Developer/2000 (ORACLE), New Era (Informix), SQL Windows (Gupta), Delphi (Borland) тощо) і генератори кодів, що входять до складу Vantage Team Builder, PRO-IV і частково — до Silverrun; засоби реінжинірингу, що забезпечують аналіз програмних кодів і схем баз даних і формування на їхній основі різних моделей і проектних специфікацій. Засоби аналізу схем БД і формування ERD входять до складу Vantage Team Builder, PRO-IV, Silverrun, Designer/2000, ERwin і S-Designor. У сфері аналізу програмних кодів найбільше поширення отримують об’єктно орієнтомовою С++ (Rational Rose (Rational Software), Object Team (Cayenne)). До допоміжних типів належать: засоби планування й управління проектом (SE Companion, Microsoft Project тощо); засоби конфігураційного управління (PVCS (Intersolv)); засоби тестування (Quality Works (Segue Software)); засоби документування (SoDA (Rational Software)). Нині вітчизняний ринок програмного забезпечення має у своєму розпорядженні такі найкраще розвинуті CASE-засоби: Vantage Team Builder (Westmount I-CASE); Designer/2000; Silverrun; ERwin+BPwin; S-Designеr; CASE-Аналітик. Крім того, на ринку постійно з’являються як нові для вітчизняних користувачів системи (наприклад, CASE /4/0, PRO-IV, System Architect, Visible Analyst Workbench, EasyCASE), так і нові версії та модифікації названих систем. Структурне моделювання Як зазначалося раніше, існують кілька підходів до автоматизованого аналізу складних систем. Вагоме місце серед них посідає структурно-функціональне моделювання, яке отримало самостійний розвиток та має досить популярні реалізації в конкретних технологіях та програмних продуктах. Структурно-функціональне моделювання започатковане у теорії автоматичного управління (ТАУ), де було розвинуто апарат, що містить не тільки правила утворення і перетворення, а й досить загальну методологію аналізу і синтезу структурних схем. Хоча динамічні структурно-функціональні схеми ТАУ мають широкі можливості для аналізу неперервних, лінійних динамічних систем, що описуються диференційними рівняннями, вони погано підходять для описування процесів у організаційних системах, де зв’язки між окремими блоками мають набагато ширший зміст і рідко можуть бути зведені до деякої функції часу (сигналу). Подальший розвиток структурно-функціонального моделювання пов’язаний із виникненням автоматизованих систем управління виробництвом (АСУ). Загалом АСУ використовують мову структурно-функціонального моделювання, яка застосовується при системному аналізі і проектуванні автоматизованих організаційних систем. Сучасні методи структурно-функціонального аналізу і моделювання складних систем були закладені завдяки працям професора Масачусетського технологічного інституту Дугласа Росса, який уперше використовував поняття «структурний аналіз» ще сорок років тому, намагаючись створити алгоритмічну мову АРТ, орієнтовану на модульне програмування. Подальший розвиток ідеї описування складних об’єктів як ієрархічних, багаторівневих, модульних систем за допомогою невеликого набору типових елементів привів до появи SADT (Structured Analyses and Design Technique), що в дослівному перекладі означає «технологія структурного аналізу і проектування», а власне кажучи, є методологією структурно-функціонального моделювання й аналізу складних систем [20]. З часу своєї появи SADT постійно удосконалювалася і широко використовувалася для ефективного вирішення цілого ряду проблем, таких як удосконалення управління фінансами та матеріально-технічним постачанням великих фірм, розробка програмного забезпечення АСУ телефонними мережами, стратегічне планування діяльності фірм, проектування обчислювальних систем і мереж тощо. Центральною ідеєю SADT за визначенням її авторів є SA-блок — універсальна одиниця універсальної пунктуації для необмеженого строго структурного аналізу. Незважаючи на таку мудровану назву, під таємничим SA-блоком ховається звичайний функціональний блок, що характеризується наявністювходу, виходу,механізмута керування. Іншим фундаментальним поняттям SADT є принцип ієрархічної декомпозиції зверху вниз, що дає можливість аналізувати якзавгодно складні системи. Оригінальним у SADT є ефективний метод кодування зв’язків, заснований на використанні спеціальних ICOM-кодів, який дає змогу не тільки спростити процедуру моделювання, але й автоматизувати процедури структурно-функціонального аналізу. Відомим програмним продуктом, що реалізує методологію структурно-функціонального аналізу SADT, є Design/IDEF виробництва компанії Meta Software Corp. Він орієнтований на проектування і моделювання складних систем широкого призначення, пов’язаних з автоматизацією і комп’ютеризацією виробництва, а також із завданнями економіко-організаційного управління та бізнес-планування. Design/IDEF має швидку і високоякісну графіку, яка уможливлює створення SADT-моделей, містить словник даних, що дає змогу зберігати необмежений обсяг інформації про об’єкти і моделі, допускає колективну роботу над моделлю, уможливлює генерування звітів за результатами системного аналізу. Першою рисою, що вирізняє SADT-методологію, є принцип побудови моделі зверху вниз. Цей принцип означає, що можна, починаючи з досить простих макроекономічних моделей розвитку сектору економіки в цілому чи окремої галузі, дійти, якщо потрібно, до окремих технологічних процесів. При цьому відповідно до призначення моделі на кожному рівні можна сформулювати обґрунтовані вимоги щодо її точності. Очевидно, що на першому етапі побудови ієрархії моделей можна та необхідно починати з досить грубих (ескізних) моделей. Оскільки методологія SADT дає змогу уточнювати (деталізувати) моделі за допомогою розкриття SADT-блоків вищого рівня ієрархії, нові штрихи за необхідності можуть бути додані без зміни тих моделей, що вже побудовані. У такий спосіб SADT реалізує ієрархічне, багаторівневе моделювання, і в цьому її друга відмінність від відомих підходів. Третьою особливістю моделювання на основі SADT є можливість одночасно зі структуруванням проблеми розробляти структуру бази даних, а точніше — баз даних, тому що на різних рівнях ієрархічного моделювання доцільно мати окремі бази даних. У пакеті DESIGN/IDEF автоматизовано процес опису бази даних, що відповідає структурі моделі. Отже, одночасно з ієрархічною структурою моделі одержують і структуру розподіленої бази даних. Для моделювання баз даних використовують мову SQL. Отже, можна висновувати, що застосування методології SADT дає змогу уніфікувати різні блоки моделі складної системи, розподілити процес створення моделі і об’єднати окремі модулі в єдину ієрархічну динамічну модель. Ще одним широко відомим інструментальним засобом структурно-функціонального моделювання, заснованим на стандарті IDEFО, є пакет BPWin, що пропонується компанією MacroProject. Він призначений для моделювання й оптимізації бізнес-процесів і автоматизує багато рутинних операцій, пов’язаних з побудовою моделей організаційних систем.

**Тема 1.3. Математичні моделі об’єктів проектування.**

Поняття моделі та моделювання. Властивості та класифікація моделей. Узагальнена методика математичного моделювання. Особливості аналітичного динамічного моделювання. Визначення оптимального розв'язку з допомогою математичних методів.

**Поняття «моделі» та «моделювання»**

Термін «модель» широко вживаний не лише в науковій літературі, причому залежно від ситуації в нього вкладається різний зміст. Слово «модель» походить від латинського «modulus», що означає міра, мірило, зразок, норма.

У найширшому сенсі під словом «модель» розуміють деякий образ об’єкта (зокрема, умовний чи уявний), що нас цікавить, або, навпаки — прообраз деякого об’єкта чи системи об’єктів. Наприклад, глобус — модель Земної кулі, фотографія — модель зображеного на ній об’єкта; карта — модель місцевості і т. ін. Що ж до розуміння моделі як прообразу, то можна згадати, наприклад, модель автомобіля, експоновану на виставці, за якою надалі поч­неться масове виготовлення таких автомобілів.

Під моделюванням розуміють дослідження об’єктів пізнання не безпосередньо, а непрямим шляхом, вивченням деяких інших допоміжних об’єктів.

Аналогією називають судження про будь-яку подібність у деякому сенсі двох об’єктів. Визначення ступеня істотності подібності чи відмінності об’єктів є умовним і відносним. Істотність подібності (відмінності) залежить від погляду спостерігача і визначається конкретною задачею. Сучасна наукова гіпотеза створюється, як правило, на підставі аналогії з виконаними на практиці науковими дослідженнями. Отже, аналогія пов’язує гіпотезу з експериментом.

Гіпотези й аналогії, що відбивають світ, який реально, об’єктив­но існує, мають бути наочними і зводитися до зручних для дослідження логічних схем. Такі логічні схеми, що спрощують міркування та логічні побудови або дають змогу проводити експерименти для уточнення природи явища, називаються моделями.

Іншими словами: модель — це деякий об’єкт-замінник об’єкта-оригіналу, що забезпечує вивчення деяких істотних, з погляду дослідника, властивостей оригіналу.

Заміщення одного об’єкта іншим із метою здобуття інформації про найважливіші властивості об’єкта-оригіналу за допомогою об’єкта-моделі називається моделюванням. Таким чином, моделювання полягає в заміні об’єкта моделлю з метою дістати інформацію про цей об’єкт, виконуючи експерименти з його моделлю. Теорія заміщення одних об’єктів (оригіналів) іншими (моделями) і дослідження властивостей об’єктів на їхніх моделях називається теорією моделювання.

Якщо результати моделювання підтверджуються і можуть бути основою для прогнозування процесів, що відбуваються в об’єкті-оригіналі, то говорять, що модель адекватна об’єктові. При цьому адекватність моделі залежить від мети моделювання і прийнятих критеріїв.

Перше й основне запитання, що постає у процесі моделювання: на якій підставі за властивостями моделі можна робити висновки про властивості об’єктів, що нас цікавлять? Єдиної відповіді на це запитання не існує. У кожному окремому дослідженні необхідно доб­ре усвідомити, на чому ґрунтується впевненість і можливість перенесення здобутих у дослідженні результатів із моделі на оригінал.

Друге запитання — а навіщо потрібно використовувати якісь допоміжні об’єкти (моделі) і навіщо розв’язувати складну проб­лему адекватності, якщо, можливо, простіше було б досліджувати об’єкт безпосередньо?

Передусім із практичних міркувань: моделі вибираються таким чином, щоб вони були значно простіші для дослідження, ніж об’єкти, що нас цікавлять. Більш того, деякі об’єкти взагалі не вдається досліджувати суто експериментально. Особливо це стосується економічних об’єктів та систем. Наприклад, пізнавальний експеримент на базі економіки будь-якої країни практично неможливий, а коли б його й здійснили, то наслідки були б дуже сумними. Окрім того, моделювання дає змогу виявляти найістотніші фактори, що впливають на поводження оригіналу.

**Основні типи моделей.**

Натурні моделі. У разі натурного моделювання оригінал (об’єкт) і модель тотожні. Такі моделі широко використовуються в техніці з метою випробувати окремі види продукції чи агрегатів: на стадії складання певну частину виробів (електронних мікросхем, двигунів, автомобілів тощо) піддають тестуванню.

Фізичні моделі. Фізичне моделювання передбачає, що об’єкт і модель мають однакову фізичну природу. Саме такими є, скажімо, літак і його геометрична модель. На цій підставі за результатами продування моделі в аеродинамічній трубі роблять висновки про аеродинамічні якості літака. Зв’язок між характеристиками літака та його моделі встановлюється згідно з теорією подібності.

Аналогові моделі. Аналогове моделювання ґрунтується на аналогії явищ, що мають різну фізичну природу, але описуються однаковими математичними рівняннями. Найпростіший приклад: вивчення механічних коливань за допомогою електричної схеми, фізичні процеси в якій описуються тими самими диференціальними рівняннями, що й коливання.

Знакові моделі. У знаковому моделюванні моделями є знакові утворення деякого виду. Розглядають вербальні моделі, схеми, графи, графіки, креслення, математичні вирази, формули тощо, причому знакові утворення та їхні елементи завжди задаються разом із тими законами (правилами), відповідно до яких ними можна оперувати.

Математичні моделі. Найважливішим видом знакового моделювання є математичне моделювання, коли моделі будуються (описуються) засобами математики і логіки.

 Под динамической системой (ДС) подразумевается любая физическая система, описываемая системой n обыкновенных дифференциальных уравнений (ОДУ) или одним ОДУ n-го по-рядка. Simulink представляет собой очень удобную среду для математического моделирования таких систем в режиме графи-ческого интерфейса.

Пусть ДС задана ОДУ n-го порядка



 где х – входная переменная (вход),

у – выходная переменная ДС (выход).

Состояние такой ДС однозначно определяется значениями n величин: y, y(1) , y(2) , y(n-1) , т.е. значениями выходной координаты и её производных до n-1 порядка включительно. Эти пе-ременные называются переменными состояния (ПС).

Если входное внешнее воздействие х отсутствует (х=0), то изменение во времени переменных состояния представляет собой свободное (собственное) движение ДС. До тех пор, пока величины, определяющие состояние системы, не изменятся во времени, система находится в равновесии. В данном случае это означает отсутствие движения и все ПС принимают нулевые значения. Если начальное (исходное) значение хотя бы одной из переменных состояния не равно нулю, то в системе возникает свободное движение. Характером свободного движения, его особенностями определяются свойства ДС.

При наличии входного воздействия (x(t) ≠ 0) в ДС возникает вынужденное движение. В теории автоматического управления входное воздействие называют задающим воздействием, поскольку оно определяет поведение и заданное значение выходной коор-динаты у.

Принципиально для программирования (разработки моделей) динамических систем можно применить два способа.

Первый способ – понижение порядка производной. Для этого уравнение необходимо разрешить относительно старшей произ-водной выходной координаты у:



По полученному выражению для старшей производной разрабатывается блок-схема модели (рис. 1). Принцип разработки блок-схемы очевиден: на входах сумматора Σ (n+1 входов) предполагаются все слагаемые выражения для . Знак алгебраического суммирования (+ или –) соответствует знаку слагаемого. Результат Σ умножается на  с помощью блока ( Gain) умножения переменной на постоянный коэффициент. На выходе этого блока формируется значение y(n). Затем выстраивается последовательная цепочка из n интеграторов (  ) для получения всех остальных производных у, включая производную нулевого порядка, т.е. саму выходную координату у. Необходимые входы сумматора формируются с помощью об-ратных связей, в которых осуществляется умножение переменных на соответствующие коэффициенты исходного уравнения.

 

 Рис. 1. Составление блок-схемы методом понижения порядка производной

Второй способ – повышения порядка производной.

Для этого исходное уравнение разрешается относительно младшей производной (в данном случае – нулевой):



По полученному выражению разрабатывается блок-схема мо-дели (рис. 2):



Рис. 2. Составление блок-схемы методом повышения порядка производной

Здесь используется тот же принцип разработки блок-схемы модели.

Формально оба способа эквивалентны. Однако есть принци-пиальная разница: основу первой схемы составляют интеграторы , основу второй схемы – дифференциаторы . Численное дифференцирование реализуется с большим трудом, и при этом могут возникать проблемы, связанные с точностью и временем вычислений. Поэтому практическое применение получил способ понижения порядка производной.

I. ОПИСАНИЕ ИНСТРУМЕНТА ЛИНЕЙНОГО АНАЛИЗА SIMULINK

1. Введение в схему модели точек входа и выхода

2. Получение характеристик в окне линейного анализа

3. Параметры графиков для Step (переходной процесс) и Bode (ЛЧХ)

II. ИССЛЕДОВАНИЕ АС 3-ГО ПОРЯДКА С ИСПОЛЬЗОВАНИЕМ ИНСТРУМЕНТАРИЯ ЛИНЕЙНОГО АНАЛИЗА (LINEAR ANALYSIS) SIMULINK

1. Модель

2. Задание

3. таблица параметров

I. ОПИСАНИЕ ИНСТРУМЕНТА ЛИНЕЙНОГО АНАЛИЗА SIMULINK

В систему Simulink входит инструмент линейного анализа АС (LTI – Linear Time-Invariant models (линейные стационарные модели)). С его помощью можно получить ряд характеристик линеаризованной модели АС: график переходного процесса h(t), график весовой функции w(t), ЛАХ и ЛФХ разомкнутой и замкнутой АС, траекторию годографа АФХ (диаграмму Найквиста) и другие.

ИССЛЕДОВАНИЕ АС 3-ГО ПОРЯДКА С ИСПОЛЬЗОВАНИЕМ ИНСТРУМЕНТАРИЯ ЛИНЕЙНОГО АНАЛИЗА (LINEAR ANALYSIS) SIMULINK

1. Модель

Имеется автоматическая система 3-го порядка, прямая цепь которой состоит из последовательного соединения колебатель-ного и апериодического звеньев. Система охвачена обратной связью с коэффициентом передачи kос (рис. 12).



**Тема 2.1.4. Математичне забезпечення комп’ютерного проектування. Декомпозиція.**

**Декомпозиція інформаційних систем**

Математичне моделювання і обчислювальні методи в задачах проектування комп’ютерних систем автоматики та управління. Інструментальні засоби моделювання. Правила декомпозиції. Модульна декомпозиція.

**Декомпозиция подсистем на модули**

Известны два типа моделей модульной декомпозиции:

* модель потока данных;
* модель объектов.

В основе модели потока данных лежит разбиение по функциям.

Модель объектов основана на слабо сцепленных сущностях, имеющих собственные наборы данных, состояния и наборы операций.

Очевидно, что выбор типа декомпозиции должен определяться сложностью разбиваемой подсистемы.

## Модульность

Модуль — фрагмент программного текста, являющийся строительным блоком для физической структуры системы. Как правило, модуль состоит из интерфейсной части и части-реализации.

Модульность — свойство системы, которая может подвергаться декомпозиции на ряд внутренне связанных и слабо зависящих друг от друга модулей.

По определению Г. Майерса, модульность — свойство ПО, обеспечивающее интеллектуальную возможность создания сколь угодно сложной программы [52]. Проиллюстрируем эту точку зрения.

Пусть С(х) — функция сложности решения проблемы х, Т(х) — функция затрат времени на решение проблемы х. Для двух проблем р1 и р2 из соотношения С(р1) > С(р2) следует, что

 T(pl)>T(p2). (4.1)

Этот вывод интуитивно ясен: решение сложной проблемы требует большего времени.

Далее. Из практики решения проблем человеком следует:

С(р1+ р2)>С(р1) + С(р2).

Отсюда с учетом соотношения (4.1) запишем:

 T(pl+p2)>T(pl) + T(p2). (4.2)

Соотношение (4.2) — это обоснование модульности. Оно приводит к заключению «разделяй и властвуй» — сложную проблему легче решить, разделив ее на управляемые части. Результат, выраженный неравенством (4.2), имеет важное значение для модульности и ПО. Фактически, это аргумент в пользу модульности.

Однако здесь отражена лишь часть реальности, ведь здесь не учитываются затраты на межмодульный интерфейс. Как показано на рис. 4.1, с увеличением количества модулей (и уменьшением их размера) эти затраты также растут.



**Рис. 4.1.** Затраты на модульность

Таким образом, существует оптимальное количество модулей Opt, которое приводит к минимальной стоимости разработки. Увы, у нас нет необходимого опыта для гарантированного предсказания Opt. Впрочем, разработчики знают, что оптимальный модуль должен удовлетворять двум критериям:

* снаружи он проще, чем внутри;
* его проще использовать, чем построить.

## Информационная закрытость

Принцип информационной закрытости (автор — Д. Парнас, 1972) утверждает: содержание модулей должно быть скрыто друг от друга [60]. Как показано на рис. 4.2, модуль должен определяться и проектироваться так, чтобы его содержимое (процедуры и данные) было недоступно тем модулям, которые не нуждаются в такой информации (клиентам).



**Рис. 4.2.** Информационная закрытость модуля

Информационная закрытость означает следующее:

1) все модули независимы, обмениваются только информацией, необходимой для работы;

2) доступ к операциям и структурам данных модуля ограничен.

Достоинства информационной закрытости*:*

* обеспечивается возможность разработки модулей различными, независимыми коллективами;
* обеспечивается легкая модификация системы (вероятность распространения ошибок очень мала, так как большинство данных и процедур скрыто от других частей системы).

Идеальный модуль играет роль «черного ящика», содержимое которого невидимо клиентам. Он прост в использовании — количество «ручек и органов управления» им невелико (аналогия с эксплуатацией телевизора). Его легко развивать и корректировать в процессе сопровождения программной системы. Для обеспечения таких возможностей система внутренних и внешних связей модуля должна отвечать особым требованиям. Обсудим характеристики внутренних и внешних связей модуля.

## Связность модуля

Связность модуля (Cohesion) — это мера зависимости его частей [58], [70], [77]. Связность — внутренняя характеристика модуля. Чем выше связность модуля, тем лучше результат проектирования, то есть тем «черней» его ящик (капсула, защитная оболочка модуля), тем меньше «ручек управления» на нем находится и тем проще эти «ручки».

Для измерения связности используют понятие силы связности (СС). Существует 7 типов связности:

1. **Связность по совпадению (СС=0).** В модуле отсутствуют явно выраженные внутренние связи.

2. **Логическая связность (СС=1).** Части модуля объединены по принципу функционального подобия. Например, модуль состоит из разных подпрограмм обработки ошибок. При использовании такого модуля клиент выбирает только одну из подпрограмм.

Недостатки:

1. сложное сопряжение;
2. большая вероятность внесения ошибок при изменении сопряжения ради одной из функций.
3. Временная связность (СС=3). Части модуля не связаны, но необходимы в один и тот же период работы системы. Недостаток: сильная взаимная связь с другими модулями, отсюда — сильная чувствительность внесению изменений.
4. Процедурная связность (СС=5). Части модуля связаны порядком выполняемых ими действий, реализующих некоторый сценарий поведения.
5. Коммуникативная связность (СС=7). Части модуля связаны по данным (работают с одной и той же структурой данных).
6. Информационная (последовательная) связность (СС=9). Выходные данные одной части используются как входные данные в другой части модуля.
7. Функциональная связность (СС=10). Части модуля вместе реализуют одну функцию.

Отметим, что типы связности 1,2,3 — результат неправильного планирования архитектуры, а тип связности 4 — результат небрежного планирования архитектуры приложения.

Общая характеристика типов связности представлена в табл. 4.1.

**Таблица 4.1.** Характеристика связности модуля

|  |  |  |
| --- | --- | --- |
| **Тип связности**  | **Сопровождаемость**  | **Роль модуля**  |
| Функциональная  |  Лучшая сопровождаемость   | «Черный ящик»  |
| Информационная ( последовательная )  | Не совсем «черный ящик»  |
| Коммуникативная  | «Серый ящик»  |
| Процедурная  |  Худшая сопровождаемость    | «Белый» или «просвечивающий ящик»  |
| Временная  |   |
| Логическая  | «Белый ящик»  |
| По совпадению  |   |

### Функциональная связность

Функционально связный модуль содержит элементы, участвующие в выполнении одной и только одной проблемной задачи. Примеры функционально связных модулей:

* Вычислять синус угла;
* Проверять орфографию;
* Читать запись файла;
* Вычислять координаты цели;
* Вычислять зарплату сотрудника;
* Определять место пассажира.

Каждый из этих модулей имеет единичное назначение. Когда клиент вызывает модуль, выполняется только одна работа, без привлечения внешних обработчиков. Например, модуль Определять место пассажира должен делать только это; он не должен распечатывать заголовки страницы.

Некоторые из функционально связных модулей очень просты (например, Вычислять синус угла или Читать запись файла), другие сложны (например, Вычислять координаты цели). Модуль Вычислять синус угла, очевидно, реализует единичную функцию, но как может модуль Вычислять зарплату сотрудника выполнять только одно действие? Ведь каждый знает, что приходится определять начисленную сумму, вычеты по рассрочкам, подоходный налог, социальный налог, алименты и т. д.! Дело в том, что, несмотря на сложность модуля и на то, что его обязанность исполняют несколько подфункций, если его действия можно представить как единую проблемную функцию (с точки зрения клиента), тогда считают, что модуль функционально связен.

Приложения, построенные из функционально связных модулей, легче всего сопровождать. Соблазнительно думать, что любой модуль можно рассматривать как однофункциональный, но не надо заблуждаться. Существует много разновидностей модулей, которые выполняют для клиентов перечень различных работ, и этот перечень нельзя рассматривать как единую проблемную функцию. Критерий при определении уровня связности этих нефункциональных модулей — как связаны друг с другом различные действия, которые они исполняют.

### Информационная связность

При информационной (последовательной) связности элементы-обработчики модуля образуют конвейер для обработки данных — результаты одного обработчика используются как исходные данные для следующего обработчика. Приведем пример:

Модуль Прием и проверка записи

* прочитать запись из файла
* проверить контрольные данные в записи
* удалить контрольные поля в записи
* вернуть обработанную запись

Конец модуля

В этом модуле 3 элемента. Результаты первого элемента (прочитать запись из файла) используются как входные данные для второго элемента (проверить контрольные данные в записи) и т. д.

Сопровождать модули с информационной связностью почти так же легко, как и функционально связные модули. Правда, возможности повторного использования здесь ниже, чем в случае функциональной связности. Причина — совместное применение действий модуля с информационной связностью полезно далеко не всегда.

### Коммуникативная связность

При коммуникативной связности элементы-обработчики модуля используют одни и те же данные, например внешние данные. Пример коммуникативно связного модуля:

Модуль Отчет и средняя зарплата

используется Таблица зарплаты служащих

сгенерировать Отчет по зарплате

вычислить параметр Средняя зарплата

вернуть Отчет по зарплате. Средняя зарплата

Конец модуля

Здесь все элементы модуля работают со структурой Таблица зарплаты служащих.

С точки зрения клиента проблема применения коммуникативно связного модуля состоит в избыточности получаемых результатов. Например, клиенту требуется только отчет по зарплате, он не нуждается в значении средней зарплаты. Такой клиент будет вынужден выполнять избыточную работу — выделение в полученных данных материала отчета. Почти всегда разбиение коммуникативно связного модуля на отдельные функционально связные модули улучшает сопровождаемость системы.

Попытаемся провести аналогию между информационной и коммуникативной связностью.

Модули с коммуникативной и информационной связностью подобны в том, что содержат элементы, связанные по данным. Их удобно использовать, потому что лишь немногие элементы в этих модулях связаны с внешней средой. Главное различие между ними — информационно связный модуль работает подобно сборочной линии; его обработчики действуют в определенном порядке; в коммуникативно связном модуле порядок выполнения действий безразличен. В нашем примере не имеет значения, когда генерируется отчет (до, после или одновременно с вычислением средней зарплаты).

### Процедурная связность

При достижении процедурной связности мы попадаем в пограничную область между хорошей сопровождаемостью (для модулей с более высокими уровнями связности) и плохой сопровождаемостью (для модулей с более низкими уровнями связности). Процедурно связный модуль состоит из элементов, реализующих независимые действия, для которых задан порядок работы, то есть порядок передачи управления. Зависимости по данным между элементами нет. Например:

Модуль Вычисление средних значений

используется Таблица-А. Таблица-В

вычислить среднее по Таблица-А

вычислить среднее по Таблица-В

вернуть среднееТабл-А. среднееТабл-В

Конец модуля

Этот модуль вычисляет средние значения для двух полностью несвязанных таблиц Таблица-А и Таблица-В, каждая из которых имеет по 300 элементов.

Теперь представим себе программиста, которому поручили реализовать данный модуль. Соблазнившись возможностью минимизации кода (использовать один цикл в интересах двух обработчиков, ведь они находятся внутри единого модуля!), программист пишет:

Модуль Вычисление средних значений

используется Таблица-А. Таблица-В

суммаТабл-А := 0

суммаТабл-В := 0

для i := 1 до 300

суммаТабл-А := суммаТабл-А + Таблица-А(i)

суммаТабл-В :- суммаТабл-В + Таблица-В(i)

конец для

среднееТабл-А := суммаТабл-А / 300

среднееТабл-В := суммаТабл-В / 300

вернуть среднееТабл-А, среднееТабл-В

Конец модуля

Для процедурной связности этот случай типичен — независимый (на уровне проблемы) код стал зависимым (на уровне реализации). Прошли годы, продукт сдали заказчику. И вдруг возникла задача сопровождения — модифицировать модуль под уменьшение размера таблицы В. Оцените, насколько удобно ее решать.

### Временная связность

При связности по времени элементы-обработчики модуля привязаны к конкретному периоду времени (из жизни программной системы).

Классическим примером временной связности является модуль инициализации:

Модуль Инициализировать Систему

перемотать магнитную ленту 1

Счетчик магнитной ленты 1 := 0

перемотать магнитную ленту 2

Счетчик магнитной ленты 2 := 0

Таблица текущих записей : = пробел..пробел

Таблица количества записей := 0..0

Переключатель 1 : = выкл

Переключатель 2 := вкл

Конец модуля

Элементы данного модуля почти не связаны друг с другом (за исключением того, что должны выполняться в определенное время). Они все — часть программы запуска системы. Зато элементы более тесно взаимодействуют с другими модулями, что приводит к сложным внешним связям.

Модуль со связностью по времени испытывает те же трудности, что и процедурно связный модуль. Программист соблазняется возможностью совместного использования кода (действиями, которые связаны только по времени), модуль становится трудно использовать повторно.

Так, при желании инициализировать магнитную ленту 2 в другое время вы столкнетесь с неудобствами. Чтобы не сбрасывать всю систему, придется или ввести флажки, указывающие инициализируемую часть, или написать другой код для работы с лентой 2. Оба решения ухудшают сопровождаемость.

Процедурно связные модули и модули с временной связностью очень похожи. Степень их непрозрачности изменяется от темного серого до светло-серого цвета, так как трудно объявить функцию такого модуля без перечисления ее внутренних деталей. Различие между ними подобно различию между информационной и коммуникативной связностью. Порядок выполнения действий более важен в процедурно связных модулях. Кроме того, процедурные модули имеют тенденцию к совместному использованию циклов и ветвлений, а модули с временной связностью чаще содержат более линейный код.

### Логическая связность

Элементы логически связного модуля принадлежат к действиям одной категории, и из этой категории клиент выбирает выполняемое действие. Рассмотрим следующий пример:

Модуль Пересылка сообщения

переслать по электронной почте

переслать по факсу

послать в телеконференцию

переслать по ftp-протоколу

Конец модуля

Как видим, логически связный модуль — мешок доступных действий. Действия вынуждены совместно использовать один и тот же интерфейс модуля. В строке вызова модуля значение каждого параметра зависит от используемого действия. При вызове отдельных действий некоторые параметры должны иметь значение пробела, нулевые значения и т. д. (хотя клиент все же должен использовать их и знать их типы).

Действия в логически связном модуле попадают в одну категорию, хотя имеют не только сходства, но и различия. К сожалению, это заставляет программиста «завязывать код действий в узел», ориентируясь на то, что действия совместно используют общие строки кода. Поэтому логически связный модуль имеет:

* уродливый внешний вид с различными параметрами, обеспечивающими, например, четыре вида доступа;
* запутанную внутреннюю структуру со множеством переходов, похожую на волшебный лабиринт.

В итоге модуль становится сложным как для понимания, так и для сопровождения.

### Связность по совпадению

Элементы связного по совпадению модуля вообще не имеют никаких отношений друг с другом:

Модуль Разные функции (какие-то параметры)

поздравить с Новым годом (...)

проверить исправность аппаратуры (...)

заполнить анкету героя (...)

измерить температуру (...)

вывести собаку на прогулку (...)

запастись продуктами (...)

приобрести «ягуар» (...)

Конец модуля

Связный по совпадению модуль похож на логически связный модуль. Его элементы-действия не связаны ни потоком данных, ни потоком управления. Но в логически связном модуле действия, по крайней мере, относятся к одной категории; в связном по совпадению модуле даже это не так. Словом, связные по совпадению модули имеют все недостатки логически связных модулей и даже усиливают их. Применение таких модулей вселяет ужас, поскольку один параметр используется для разных целей.

Чтобы клиент мог воспользоваться модулем Разные функции, этот модуль (подобно всем связным по совпадению модулям) должен быть «белым ящиком», чья реализация полностью видима. Такие модули делают системы менее понятными и труднее сопровождаемыми, чем системы без модульности вообще!

К счастью, связность по совпадению встречается редко. Среди ее причин можно назвать:

* бездумный перевод существующего монолитного кода в модули;
* необоснованные изменения модулей с плохой (обычно временной) связностью, приводящие к добавлению флажков.

### Определение связности модуля

Приведем алгоритм определения уровня связности модуля.

1. Если модуль — единичная проблемно-ориентированная функция, то уровень связности — функциональный; конец алгоритма. В противном случае перейти к пункту 2.

2. Если действия внутри модуля связаны, то перейти к пункту 3. Если действия внутри модуля никак не связаны, то перейти к пункту 6.

3. Если действия внутри модуля связаны данными, то перейти к пункту 4. Если действия внутри модуля связаны потоком управления, перейти к пункту 5.

4. Если порядок действий внутри модуля важен, то уровень связности — информационный. В противном случае уровень связности — коммуникативный. Конец алгоритма.

5. Если порядок действий внутри модуля важен, то уровень связности — процедурный. В противном случае уровень связности — временной. Конец алгоритма.

6. Если действия внутри модуля принадлежат к одной категории, то уровень связности — логический. Если действия внутри модуля не принадлежат к одной категории, то уровень связности — по совпадению. Конец алгоритма.

Возможны более сложные случаи, когда с модулем ассоциируются несколько уровней связности. В этих случаях следует применять одно из двух правил:

* правило параллельной цепи. Если все действия модуля имеют несколько уровней связности, то модулю присваивают самый сильный уровень связности;
* правило последовательной цепи. Если действия в модуле имеют разные уровни связности, то модулю присваивают самый слабый уровень связности.

Например, модуль может содержать некоторые действия, которые связаны процедурно, а также другие действия, связные по совпадению. В этом случае применяют правило последовательной цепи и в целом модуль считают связным по совпадению.

## Сцепление модулей

Сцепление (Coupling) — мера взаимозависимости модулей по данным [58], [70], [77]. Сцепление — внешняя характеристика модуля, которую желательно уменьшать.

Количественно сцепление измеряется степенью сцепления (СЦ). Выделяют 6 типов сцепления.

1. **Сцепление по данным** (СЦ=1). Модуль А вызывает модуль В.

Все входные и выходные параметры вызываемого модуля — простые элементы данных (рис. 4.13).



**Рис. 4.13.** Сцепление поданным

2. **Сцепление по образцу** (СЦ=3). В качестве параметров используются структуры данных (рис. 4.14).



**Рис. 4.14.** Сцепление по образцу

3. **Сцепление по управлению** (СЦ=4). Модуль А явно управляет функционированием модуля В (с помощью флагов или переключателей), посылая ему управляющие данные (рис. 4.15).



**Рис. 4.15.** Сцепление по управлению

4. **Сцепление по внешним ссылкам** (СЦ=5). Модули А и В ссылаются на один и тот же глобальный элемент данных.

5. **Сцепление по общей области** (СЦ=7). Модули разделяют одну и ту же глобальную структуру данных (рис. 4.16).

6. **Сцепление по содержанию** (СЦ=9). Один модуль прямо ссылается на содержание другого модуля (не через его точку входа). Например, коды их команд перемежаются друг с другом (рис. 4.16).



**Рис. 4.16.** Сцепление по общей области и содержанию

На рис. 4.16 видим, что модули В и D сцеплены по содержанию, а модули С, Е и N сцеплены по общей области.

## Сложность программной системы

В простейшем случае сложность системы определяется как сумма мер сложности ее модулей. Сложность модуля может вычисляться различными способами.

Например, М. Холстед (1977) предложил меру длины N модуля [33]:

*N ≈ n1log2 (n1) + n2log2(n2),*

где *n1* — число различных операторов, *п2 —* число различных операндов.

В качестве второй метрики М. Холстед рассматривал объем *V* модуля (количество символов для записи всех операторов и операндов текста программы):

*V = N* x *log2 (n1* + *n2).*

Вместе с тем известно, что любая сложная система состоит из элементов и системы связей между элементами и что игнорировать внутрисистемные связи неразумно.

Том МакКейб (1976) при оценке сложности ПС предложил исходить из топологии внутренних связей [49]. Для этой цели он разработал метрику цикломатической сложности:

*V(G) = E-N + 2,*

где *Е —* количество дуг, a*.N —* количество вершин в управляющем графе ПС. Это был шаг в нужном направлении. Дальнейшее уточнение оценок сложности потребовало, чтобы каждый модуль мог представляться как локальная структура, состоящая из элементов и связей между ними.

Таким образом, при комплексной оценке сложности ПС необходимо рассматривать меру сложности модулей, меру сложности внешних связей (между модулями) и меру сложности внутренних связей (внутри модулей) [28], [56]. Традиционно со внешними связями сопоставляют характеристику «сцепление», а с внутренними связями — характеристику «связность».

Вопросы комплексной оценки сложности обсудим в следующем разделе.

Про системність об’єктів свідчить те, що їх можна поділяти, оскільки лише вони мають структуру. Процеси декомпозиції й композиції є засобами отримання інформації для здійснення аналізу та синтезу систем.

Декомпозиція – це процес поділу систем на елементи, зручні для якихось операцій з нею, а саме поділ до елементів, які приймаються за неподільні об’єкти.

Будь яка система по-своєму складна. Це означає що всю сукупність інформації, яка характеризує систему, і всю сукупність зв`язків між елементами системи, неможливо сприйняти в цілому і повністю, звідси, додержуючись методу декомпозиції, для швидкого впровадження ІС необхідно дотримуватися принципу “добре структурованної системи”, і тому головна мета декомпозиції – поділ системи на простіші частини. Зменшуючи складність системи, ми забезпечуємо умови для аналізу та синтезу компонентів, для проектування, побудови, впровадження, експлуатації та вдосконалення систем управління. Поділ звичайно виконують у такий спосіб, щоб компоненти піддавались якій-небудь класифікації. Рекомендується зважати на природну декомпозицію, відбиту в існуючій структурі управління, обов’язках посадових осіб, діючого документообігу і т.п. Доцільно проводити багаторазову декомпозицію по кількох різних напрямках.

Загальна мета, критерії функціонування та основні обмеження на роботу системи звичайно формуються на початку створення системи.

Так, при декомпозиції можуть застосовуватись різні засоби, методи та ознаки поділу системи. Поділ може мати матеріальну, функціональну, алгоритмічну та іншу основу. Однак сам процес декомпозиції кінцевий, оскільки поділ відбувається до створення елементів, які приймаються за неподільні об’єкти.

Так, при поділі системи на компоненти можемо мати різні варіанти. Компонент — це частина ІС, яку після декомпозиції можемо розглядати як самостійне ціле (ГОСТ 34.003–90).

В ОРММ пропонують поділ системи здійснювати відповідно до адміністративного поділу системи керування економічним об’єктом. При такій декомпозиції виділяють: керування технічною підготовкою виробництва, техніко-економічне планування, оперативне керування виробництвом і т. ін. Також систему можна поділяти за функціями, які виконуються (облік, контроль, планування і т.п.), і за ресурсами (матеріальні, трудові, основні засоби, готова продукція, грошові).

Наступним кроком декомпозиції є виділення в компоненті функціо­нальних процесів (задач). Задача ІС, функція чи частина функції ІС, є формалізована сукупність автоматизованих дій, в результаті виконання яких здобувають результати заданого виду (ГОСТ 34.003–90). Може виявитися, що при одному й тому самому засобі декомпозиції системи на компоненти одна й та сама задача за змістом в різних проектах належить до різних компонентів. Однак неоднозначність закінчується, тільки-но процес декомпозиції доводиться до рівня економічних показників; його можемо вважати неподільним елементом, оскільки поділ його на атрибути приводить до втрати економічної суті, й він уже не зможе відігравати роль змінної, яка характеризує стан об’єкта, котрий він описує. В інформа­ційному аспекті показник не є кінцевим елементом і може бути поділений на атрибути.

У свою чергу в лінгвістичному аспекті атрибути також не є кінцевими елементами, оскільки можуть бути поділені на окремі слова та символи.

Отже, вибір основи та межі декомпозиції визначається суттю об’єкта, який досліджується, метою, предметною областю обстеження, запасом знань дослідника відносно об’єкта обстеження.

Але при поділі системи на різні рівні ієрархії потрібно виконувати наступні вимоги:

кожен рівень ієрархії повинен повністю оглядатися і бути зрозумілим без детального знання нижчіх рівнів;

зв`язки між елементами на одному рівні ієрархії мають бути мінімальними;

не повинно бути зв`язків між елементами через один рівень ієрархії;

елемент вищого рівня має викликати елемент наступного рівня і передаючи йому необхідну вхідну інформацію, повинен утворювати з ним єдине ціле;

елемент наступного рівня після закінчення своєї роботи повертає управління елементу, що його викликав.

Аналізуючи та описуючи системи, використовують такі види структур, які різняться типами елементів і зв’язками між ними (РД 50–680–88 АС «Основные положения»).

1. Функціональні (елементи – компоненти, функції, задачі, про-цедури; зв’язки – інформаційні).

2. Технічні (елементи – пристрої, компоненти, комплекси; зв’язки – лінії та канали зв’язку).

3. Організаційні (елементи – колективи людей та окремі виконавці; зв’язки – інформаційні, співпідпорядкування та взаємодії).

4. Програмні (елементи – програмні модулі та вироби; зв’язки – керуючі).

5. Інформаційні (елементи – форми існування та подання інформації в системі; зв’язки – операції перетворення інформації в системі).

6. Алгоритмічні (елементи – алгоритми; зв’язки – інформаційні).

7. Документальні (елементи – неподільні складові і документи ІС; зв’язки – взаємодії, входження і співпідпорядкування).

**Тема 2.1. Інтегровані системи автоматизованого проектування конструкцій (CAD/CAE системи).**

Автоматизоване проектування (computer-aided design — CAD). Автоматизоване конструювання (computer-aided engineering — САЕ).

Компоненти систем CAD/CAE.

**1.1. Основні поняття процесу проектування**

Сучасні підприємства не зможуть існувати в умовах ринкової економіки, якщо не випускатимуть нові продукти кращої якості, нижчої вартості і за менший час. Тому для цього їм потрібно використовувати сучасні комп’ютерні технології, їх високу швидкодію і можливості зручного графічного інтерфейсу для того, щоб автоматизувати і пов’язати один з одним завдання проектування і виробництва. Таким чином скорочується час і вартість розробки і випуску продукції. Для цієї мети використовуються технології автоматизованого проектування (computer-aided design – CAD), автоматизованого виробництва (computer-aided manufacturing – САМ) і автоматизованої розробки або конструювання (computer-aided engineering – САЕ). Щоб зрозуміти значення систем CAD/CAM/CAE необхідно вивчити різні завдання і операції, які вирішуються у процесі розробки і виробництва продукції. Всі ці завдання, узяті разом, називаються **життєвим циклом виробу** (product cycle). Приклад життєвого циклу виробу приведений на рисунку 1.1.



Рис. 1.1. Життєвий цикл виробу

Прямокутники, намальовані суцільними лініями, представляють два головні процеси, складових життєвого циклу виробу: процес розробки і процес виробництва. **Процес розробки** починається із запитів споживачів, які обслуговуються відділом маркетингу і закінчується повним описом продукту, що зазвичай виконується у формі рисунка. **Процес виробництва** починається з технічних вимог і закінчується постачанням готових виробів.

Операції, що відносяться до процесу розробки можна розділити на **аналітичні і синтетичні**. Як видно з рисунка 1.1 первинні операції розробки, такі як визначення необхідності розробки, формулювання технічних вимог, аналіз здійсненності і збір важливої інформації, а також концептуалізація розробки відносяться до підпроцесу синтезу. Результатом підпроцесу синтезу є концептуальний проект передбачуваного продукту у формі ескізу або топологічного креслення, що відображає зв’язки різних компонентів продукту. У цій частині циклу робляться основні фінансові вливання, необхідні для реалізації ідеї виробу, а також визначається його функціональність.

Велика частина інформації, що утворюється і обробляється у рамках підпроцесу синтезу є якісною, а отже, незручною для комп’ютерної обробки.

Готовий концептуальний проект аналізується і оптимізується – це **підпроцес аналізу**. Перш за все виробляється аналітична модель, оскільки аналізується саме модель, а не проект. Не дивлячись на швидке зростання кількості і якості комп’ютерів, використовуваних в конструюванні, у майбутньому відмовлятися від використання абстракції аналітичної моделі не доцільно. Аналітичне моделювання відбувається, якщо з проекту видалити несуттєві елементи, редукувати розмірності і врахувати наявну симетрію. Для прикладу, редукція розмірностей – це заміна тонкого листа з будь-якого матеріалу на еквівалентну площину з атрибутом товщини, або довгої і тонкої ділянки на лінію з певними параметрами, що характеризують поперечний розріз. Симетричність геометрії тіла і навантаження, прикладені до нього, дозволяє розглядати в моделі лише певну її частину.

Типові приклади аналізу: аналіз напруження, що дозволяє перевірити міцність конструкції, контроль зіткнень, що виявляє можливість зіткнень рухомих частин, складових механізмів, а також кінематичний аналіз, що показує, як проектований пристрій здійснюватиме очікувані рухи. Якість результатів, які можуть бути одержані при аналізі, безпосередньо пов’язані з якістю вибраної аналітичної моделі, якою вони обмежуються.

Після завершення проектування і вибору оптимальних параметрів починається **етап оцінки проекту.** Для цієї мети можуть виготовлятися прототипи. У конструюванні прототипів великої популярності набуває нова технологія швидкого прототипування (rapid prototyping). Ця технологія дозволяє конструювати прототип від низу до верху, тобто безпосередньо з проекту, оскільки фактично вимагаються тільки дані про поперечний перетин конструкції. Якщо оцінка проекту на підставі прототипу показує, що проект не задовольняє вимогам, розглянутий процес розробки повторюється знову.

Якщо результат оцінки проекту виявляється задовільним, починається **підготовка проектної документації.** До неї відносяться креслення, звіти і списки матеріалів. Креслення зазвичай копіюються, а копії передаються на виробництво.

Як видно з рисунка 1.1, процес виробництва починається з планування, яке виконується на підставі одержаних на етапі проектування креслень, а закінчується готовим продуктом (виробом). **Технологічна підготовка виробництва** – це операція, що встановлює список технологічних процесів по виготовленню продукту із заданими параметри. Одночасно вибирається устаткування, на якому проводитимуться технологічні операції, такі як отримання деталі потрібної форми із заготівки. В результаті підготовки виробництва складаються плани випуску, списки матеріалів і програми для устаткування. На цьому ж етапі обробляються інші специфічні вимоги, зокрема розглядаються конструкції затискачів і кріплень. У процесі виробництва підготовка займає таке ж місце, як синтез у процесі проектування, вимагаючи при цьому значного людського досвіду і ухвалення якісних рішень. У цьому випадку вимагається значної комп’ютеризації даного етапу. Після завершення технологічної підготовки починається випуск готового продукту і його перевірка на відповідність вимогам. Якісні елементи виробу, що успішно пройшли контроль, збираються разом, проходять тестування функціональності, упаковуються, маркуються і відвантажуються замовникам.

Розглянемо, як для описаного вище типового життєвого циклу виробу можуть бути застосовані технології CAD, САМ і САЕ. Комп’ютери не можуть широко використовуватися у підпроцесі синтезу, оскільки вони не володіють здатністю якісно обробляти інформацію. Проте навіть на цьому етапі розробник може, за допомогою комерційних баз даних успішно збирати важливу для аналізу інформацію, а також користуватися даними з каталогів.

Але використання комп’ютерів у процесі концептуалізації проекту, не має широкого застосування, тому що комп’ютер ще не став засобом для інтелектуальної творчості. На цьому етапі комп’ютери можуть застосовуватися для забезпечення ефективності створення різних концептуальних проектів. Для цього також можна застосовувати засоби параметричного і геометричного моделювання, а також макропрограми в системах автоматизованої розробки креслень (computer-aided drafting або САD). **Система геометричного моделювання** (geometric modeling system) – це тривимірний еквівалент системи автоматизованої розробки креслень, тобто програмний пакет, що працює з тривимірними об’єктами.

У аналітичній фазі проектування комп’ютери здобули широкого застосування. Програмних пакетів для аналізу напружень, контролю зіткнень і кінематичного аналізу існує дуже багато. Ці програмні пакети відносяться до засобів автоматизованого конструювання (САЕ). Головна проблема, пов’язана з їх використанням полягає в необхідності формування аналітичної моделі. Проте, аналітична модель не ідентична концептуальному проекту – вона виводиться з нього шляхом виключення неістотних деталей і редукції розмірностей. Необхідний рівень абстракції залежить від типу аналізу і бажаної точності рішення. Отже, автоматизувати процес абстрагування досить складно, тому аналітичну модель часто створюють окремо.

Зазвичай абстрактна модель проекту створюється в системі розробки робочих креслень або в системі геометричного моделювання, а іноді за допомогою засобів аналітичного пакету. Аналітичні пакети потребують, щоб досліджувана структура була представлена у вигляді об’єднання окремих ланок (сіток), що розділяють об’єкт на окремі ділянки, зручні для комп’ютерної обробки. Якщо аналітичний пакет може генерувати сітку автоматично, конструктору залишається задати тільки межі абстрактного об’єкту. Процес створення сітки називається моделюванням методом кінцевих елементів (finite-element modeling). Моделювання цим методом включає також завдання граничних умов і зовнішніх навантажень.

Підпроцес аналізу може виконуватися у поєднанні з оптимізацією проекту за відповідними параметрами, застосовуючи алгоритми пошуку оптимальних рішень. Процес оптимізації може входити до загальної системи автоматизованого проектування, але доцільно здійснювати оптимізацію окремо.

Фаза оцінки проекту потребує виготовлення прототипу, який можна сконструювати за допомогою програмних пакетів швидкого прототипування. Такі пакети вважаються програмами для автоматизованої підготовки виробництва (САМ). Дані, що визначають форму прототипу, виходять у результаті геометричного моделювання. Швидке прототипування – зручний спосіб конструювання прототипу, проте у сучасних умовах за допомогою відповідних програмних засобів зручніше користуватися віртуальним прототипом, який часто називається «цифровою копією» (digital mock-up), що дає змогу одержати такі ж дані, що і реальний прототип.

Побудова цифрової копії називається віртуальним прототипуванням. **Віртуальний прототип** може бути створений у спеціалізованій програмі геометричного моделювання.

Остання фаза процесу розробки – **підготовка проектної документації**. На цьому етапі надзвичайно корисним виявляється використання систем підготовки робочих креслень.

**Процес виробництва** включає планування випуску, проектування і придбання нових інструментів, замовлення матеріалів, програмування машин з ЧПУ, контроль якості і упаковування. Комп’ютерні системи, які використовуються у цих операціях, можуть бути класифіковані як системи автоматизованого виробництва. Наприклад, програма автоматизованої технологічної підготовки (computer-aided process planning – САРР) використовується на етапі підготовки виробництва і відноситься до систем автоматизованого виробництва (САМ). Процес підготовки виробництва складно автоматизувати, тому повністю автоматичних систем технологічної підготовки не існує. Проте існують програмні пакети, що генерують код для верстатів з числовим програмним управлінням. Верстати цього класу дозволяють одержати деталь потрібної форми за даними, які надходять з ЕОМ. До систем автоматизованого виробництва відносять також програмні пакети, керування рухом роботів при збірці компонентів і переміщенні їх між операціями, а також пакети, що дозволяють програмувати координатно-вимірювальну машину (coordinate measuring machine – СММ).

**1.2. Визначення CAD, САМ і САЕ**

**Автоматизоване проектування** (computer-aided design – CAD) є технологією суть якої полягає у використанні комп’ютерних систем для полегшення створення, змін, аналізу і оптимізації проектів. Таким чином, будь-яка програма, що працює з комп’ютерною графікою, так само як і будь-який додаток використовуваний в інженерних розрахунках, відноситься до систем автоматизованого проектування. До складу засобів CAD відносять геометричні програми для роботи з формами, а також спеціалізовані додатки для аналізу і оптимізації. До цих засобів також відносять програми для аналізу допусків, розрахунку масо-інерційних властивостей, моделювання методом кінцевих елементів і візуалізації результатів аналізу. Найголовніша функція CAD – визначення геометрії конструкції (деталі механізму, архітектурні елементи, електронні схеми, плани будівель і т. п.), оскільки геометрія визначає всі подальші етапи життєвого циклу виробу. Для цієї мети зазвичай використовуються системи розробки робочих креслень і геометричного моделювання. Тому ці системи вважаються системами автоматизованого проектування.

Крім того, геометричні параметри, які визначені у цих системах можуть використовуватися як основа для подальших операцій в системах САЕ і САМ. До переваг засобів CAD відносять: економія часу, скорочення кількості помилок пов’язаних з необхідністю визначати геометрію конструкції. Таким чином, системи автоматизованої розробки робочих креслень і системи геометричного моделювання є найбільш важливими компонентами автоматизованого проектування.

**Автоматизоване виробництво** (computer-aided manufacturing – САМ) – це технологія, що полягає у використанні комп’ютерних систем для планування, управління і контролю операцій виробництва через прямий або непрямий інтерфейс з виробничими ресурсами підприємства. Одним з найбільш широко застосовуваних підходів до автоматизації виробництва є числове програмне управління (ЧПУ, numerical control – NC). ЧПУ полягає у використанні запрограмованих команд для управління верстатом, який може шліфувати, різати, фрезерувати, штампувати, згинати та іншими способами перетворювати заготовки на готові деталі. У наш час комп’ютери здатні генерувати великі програми для верстатів з ЧПУ на підставі геометричних параметрів виробів з бази даних CAD і додаткових відомостей, що надаються оператором. Напрям розвитку у цій галузі полягає у скороченні необхідності втручання оператора.

Ще одна важлива функція систем автоматизованого виробництва – програмування роботів, які можуть працювати на гнучких автоматизованих ділянках, вибираючи і встановлюючи інструменти проводячи обробку на верстатах з ЧПУ. Роботи можуть виконувати власні завдання, наприклад зварювання, збірка і перенесення устаткування і деталей по цеху.

Планування процесів може визначати послідовність операцій по виготовленню пристрою від початку і до кінця на всьому необхідному устаткуванні. Хоча повністю автоматизоване планування процесів практично неможливе, але план обробки конкретної деталі цілком може бути сформований автоматично, якщо вже є плани обробки аналогічних деталей. Для цього була розроблена технологія групування, що дозволяє об’єднувати схожі деталі в сімейства. Деталі вважаються подібними, якщо вони мають загальні виробничі особливості (вузли, пази, отвори і т. д.). Для автоматичного виявлення схожості деталей необхідно, щоб база даних CAD містила відомості про такі особливості. Це завдання здійснюється за допомогою об’єктно-орієнтованого моделювання або розпізнавання елементів.

**Автоматизоване конструювання** (computer-aided engineering – САЕ) – полягає у використанні комп’ютерних систем для аналізу геометрії CAD, моделювання і вивчення поведінки виробу для удосконалення і оптимізації його конструкції. Засоби САЕ можуть здійснювати багато різних варіантів аналізу. Програми для кінематичних розрахунків, здатні визначати траєкторії руху і швидкості ланок в механізмах. Програми динамічного аналізу можуть використовуватися для визначення навантажень і зсувів в складних пристроях типу автомобілів. Програми верифікації і аналізу логіки і синхронізації імітують роботу складних електронних ланцюгів.

Зі всіх методів комп’ютерного аналізу найширше в конструюванні використовується метод кінцевих елементів (finite-element method – FEM). З його допомогою розраховуються напруженість, деформації, теплообмін, розподіл магнітного поля, потоки рідин та інші завдання з безперервними середовищами, вирішувати які будь-яким іншим методом є непрактично. У методі кінцевих елементів аналітична модель структури є з’єднанням елементів, завдяки чому вона розбивається на окремі частини, які вже можуть оброблятися комп’ютером.

Крім згаданих вище засобів існує багато програмних засобів для оптимізації конструкцій. Хоча засоби оптимізації можуть бути віднесені до класу САЕ, зазвичай їх розглядають окремо. У цих підходах початкова форма конструкції передбачається простою, як, наприклад, у прямокутного двовимірного об’єкту, що складається з невеликих елементів різної щільності. Потім виконується процедура оптимізації, що дозволяє визначити конкретні значення щільності та досягти певної мети з урахуванням обмежень на напруженість.

Після визначення оптимальних значень щільності розраховується оптимальна форма об’єкту. Перевагами методів аналізу і оптимізації конструкцій є те, що вони дозволяють конструктору побачити поведінку кінцевого виробу і виявити можливі помилки до створення і тестування реальних прототипів, уникнувши певних витрат. Оскільки вартість конструювання на останніх стадіях розробки і виробництва продукту є значною, а це призводить до скорочення термінів і вартості розробки.

Таким чином, технології CAD, САМ і САЕ полягають в автоматизації і підвищенні ефективності конкретних стадій життєвого циклу виробу. Розвиваючись незалежно, ці системи потребують інтеграції процесів проектування і виробництва. Для вирішення цієї проблеми була запропонована нова технологія, що одержала назву комп’ютерно-інтегрованого виробництва (computer-integrated manufacturing – CIM). **Технологія CIM** передбачає використання комп’ютерної бази даних для ефективнішого управління всім підприємством, зокрема бухгалтерією, плануванням, доставкою і іншими завданнями, а не тільки проектуванням і виробництвом, які охоплювалися системами CAD, САМ і САЕ. CIM часто називають філософією бізнесу, а не комп’ютерною системою.

**1.3. Структура САПР**

**Структура САПР**. Узагальнено структуру САПР можна представити у вигляді функціональної і забезпечуючої частин (рис. 1.2).



Рис. 1.2. Структурна схема САПР

**Функціональна частина САПР** на схемі представлена у вигляді набору підсистем, що вирішують питання проектування: технологічної підготовки виробництва, моделювання, інформаційного пошуку, інженерних розрахунків, управління, випробувань, виготовлення, машинної графіки. Підсистеми є основними структурними ланками САПР і розрізняються за призначенням і по відношенню до об’єкту проектування.

Кожна із підсистем САПР може бути визначена як комплекс програмних засобів, призначених для виконання певного процесу проектування, а програмні компоненти взаємозв’язані з технічними засобами САПР.

Існуючий вітчизняний і зарубіжний досвід у галузі автоматизації проектування свідчить про те, що розробка, впровадження і ефективне використання програмних комплексів призначених для автоматизації процесу проектування реалізованих на базі сучасних ЕОМ вимагають комплексного рішення широкого спектру питань: організаційних, технічних, математичних, програмних, лінгвістичних, інформаційних і ін. Вирішення цих проблем базується на відповідних видах забезпечення. Складність розробок великих комплексів взаємозв’язаних програм полягає в тому, що ефективність вирішення кожної конкретної проблеми, визначається на завершальному етапі роботи, коли вся або більша частина системи починає функціонувати; це зумовлює складність створення високоефективних програмних комплексів при первинній розробці. Система стає дієвою тільки у процесі створення, випробування і удосконалення.

**Підсистема інформаційного пошуку** – це комплекс мовноалгоритмічних засобів, призначений для зберігання і пошуку у певній множині елементів (документів, стандартів, норм, креслень виконаних конструкцій, патентів, характеристик матеріалів і т.п.) та представлення інформації, що відповідає на запит.

**Підсистема інженерних розрахунків** разом з **підсистемою машинної графіки**зазвичай застосовується на початковому етапі створення САПР і є сукупністю програмних засобів, призначених для виконання різних розрахунків (геометричних, міцністних та ін.) у режимі діалогу «людина-машина». Робота розвиненої підсистеми інженерних розрахунків тісно пов’язана з використанням різного роду математичних моделей проектованих об’єктів або процесів, для автоматизованого отримання яких призначена **підсистема моделювання**. Більшість сучасних САПР, крім обчислювальних, володіє широким спектром можливостей для введення, обробки, зберігання і виведення графічної інформації, що реалізуються програмними засобами підсистеми машинної графіки.

**Підсистема випробувань** є комплексом програмних засобів, призначених для створення програм управління випробувальним устаткуванням, обробки результатів випробувань, проведення «чисельного експерименту» з використанням математичних моделей об’єкту проектування і процесу його навантаження. Чисельний експеримент дуже важливий в процесі проектування, оскільки дозволяє визначити властивості проектованого об’єкту без виготовлення дослідних зразків і дає змогу відразу відмовитися від безперспективних варіантів, що значно зменшує витрати часу і матеріальних засобів на створення об’єкту.

**Підсистема виготовлення** призначена підготувати програми для верстатів і автоматичних ліній з числовим програмним управлінням.

**Підсистема технологічної підготовки виробництва**, як правило, виходить за рамки САПР і є самостійною системою (АСТПВ – автоматизована система технологічної підготовки виробництва).

**Підсистема управління** призначена для об’єднання роботи інших підсистем на різних етапах процесу проектування і виконання функцій координатора у колективному процесі ухвалення рішень.

Не обов’язково в кожній САПР повинен бути представлений весь набір функціональних підсистем – вони можуть поєднуватися довільно залежно від завдань, що стоять перед системою. Необхідно відзначити, що всі функціональні підсистеми тісно взаємозв’язані, тому часто неможливо провести між ними чіткі межі.

Так підсистема машинної графіки може видавати результати у вигляді програми для устаткування з числовим програмним управлінням, що безпосередньо пов’язує її з підсистемами технологічної підготовки виробництва і виготовлення. А підсистема машинної графіки може видавати замість креслень розкрій листового матеріалу, що поєднує її з підсистемою виготовлення.

Деякі підсистеми залежно від ступеня їх розвитку або призначення можуть існувати як самостійні системи (наприклад, інформаційно-пошукові системи, графічні та ін.).

**Забезпечуюча частина** – це технічні засоби і документи на машинних і інших інформаційних носіях, які необхідні у процесі проектування.

На відміну від функціональної забезпечуюча частина повинна входити в систему всіма своїми компонентами навіть у разі різного ступеня досконалості кожної з них. За відсутності будь-якої складової забезпечуючої частини не можна говорити про існування САПР у цілому, оскільки всі компоненти тісно взаємозв’язані.

**Технічне забезпечення САПР** є сукупністю взаємозв’язаних і взаємодіючих технічних засобів, що включають ЕОМ і працюючі під її управлінням зовнішні пристрої, призначені для виконання автоматизованого проектування. Технічне забезпечення ділиться на групи засобів програмної обробки даних (процесори і запам’ятовуючі пристрої у яких реалізуються перетворення даних і програмне управління обчисленнями), підготовки, введення і відображення даних (для обміну даними між користувачем і ЕОМ), виводу, зберігання і передачі даних (запам’ятовуючі, друкуючі та інші графічні пристрої і засоби зв’язку з віддаленими терміналами).

**Математичне забезпечення САПР** об’єднує опис математичних моделей проектованих об’єктів і математичних методів, реалізованих у САПР. Елементи математичного забезпечення надзвичайно різноманітні. Серед них є інваріантні елементи; до них відносяться принципи побудови функціональних моделей, методи чисельного рішення алгебраїчних і диференціальних рівнянь, постановки і вирішення задач на визначення екстремуму та ін.

**Програмне забезпечення САПР** є описом алгоритмів проектування, використаних у САПР, а також документів з вихідними текстами програм, програмами на машинних носіях і експлуатаційними документами.

**Інформаційне забезпечення САПР** об’єднує різні дані, необхідні для виконання автоматизованого проектування, які можуть бути представлені у вигляді документів на різних носіях, що містять відомості довідкового характеру про матеріали, комплектуючі вироби, типові проектні рішення, параметри елементів, зведення про стан поточних розробок у вигляді проектних рішень, параметрів проектованих об’єктів і т.п.

**Лінгвістичне забезпечення САПР** представлено сукупністю мов, які застосовуються для опису процедур автоматизованого проектування і проектних рішень, а також мовами програмування.

**Методичне забезпечення САПР** складають документи, що містять правила проектування в даній системі.

**Організаційне забезпечення САПР** включає положення, інструкції, накази, штатні розклади, кваліфікаційні вимоги і інші документи, що регламентують організаційну структуру підрозділів проектної організації і взаємодію підрозділів з комплексом засобів автоматизованого проектування.

**1.4. Класифікація САПР**

Класифікувати САПР можна за такими ознаками:

      по ступеню формалізації вирішуваних задач;

      по функціональному призначенню;

      по спеціалізації;

      по технічній організації.

**По ступеню формалізації** вирішуваних задач САПР можуть бути побудовані на вирішенні:

      повністю формалізованих задач;

      частково формалізованих задач;

      не формалізованих задач.

Системи побудовані на рішенні задач, що повністю формалізуються, для проектування складних конструкцій зазвичай не придатні, оскільки математичні моделі об’єктів проектування і процесів їх функціонування настільки складні, що повний і точний їх математичний опис на сьогоднішній день неможливий. Такі системи можуть застосовуватися тільки для вирішення простих задач проектування.

Системи побудовані на вирішенні задач, що не формалізуються, в даний час знаходяться у стадії досліджень і розробки («штучний інтелект») і для проектування також не застосовуються.

Необхідно відзначити, що в обох випадках процес проектування відбувається без втручання людини. Таким чином мова йде не про системи автоматизованого, а автоматичного проектування.

Для вирішення завдань у багатьох галузях промислового виробництва у даний час придатні тільки системи, побудовані на рішенні задач, що частково формалізуються.

Безумовно, частина завдань, пов’язаних з проектуванням деяких простих елементів конструкції може бути вирішена з використанням автоматичного проектування, але для проектування складних агрегатів і систем сьогодні повна автоматизація неможлива. Крім того, якщо йдеться про такі поняття, як форма пристрою, деталі інтер’єру, то на їх конструкцію крім функціональних вимог (аеродинамічні властивості, ергономіка, безпека) впливають і суб’єктивні чинники, наприклад мода, що також неможливо описати мовою математичних залежностей.

**За функціональним призначенням** САПР поділяються в залежності від вирішуваних задач, визначених складом функціональної частини системи:

      розрахунково-оптимізаційні;

      графічні;

      графоаналітичні;

      інформаційні і т.п.

**За спеціалізацією САПР** поділяють на спеціалізовані та інваріантні. Оскільки завдання автоматизованого проектування дуже складні, то, як правило, САПР є спеціалізовані системи, що створюються для вирішення вузьких завдань однієї галузі.

**За технічною організацією САПР** бувають однорівневі, побудовані на базі однієї достатньо продуктивної ЕОМ з набором необхідних периферійних пристроїв, і багаторівневі, такі, що включають крім базової ЕОМ ряд підпорядкованих їй автоматизованих робочих місць (АРМ), що побудовані на основі ЕОМ нижчого рівня.

**Тема 2.2. Технологічні процеси інформаційних технологій проектування (CAM та інші системи).**

Метод кінцевих елементів (finite-element method — FEM). Автоматизоване виробництво (computer-aided manufacturing — САМ). Компоненти систем CAM.

Найбільш широке застосування в конструюванні має *метод кінцевих елементів (finite-element method - FEM).*З його допомогою розраховуються потужність, деформації, теплообмін, розподілення магнітного поля, потоки рідини та інші задачі, рішати які будь-яким іншим методом непрактично. В методі кінцевих елементів аналітична модель структури представляє собою з’єднання елементів, завдяки чому вона розбивається на окремі частини, які вже можуть оброблятися комп’ютером.

Для використання методу кінцевих елементів потрібна абстрактна модель належного рівня, а не сама конструкція. Абстрактна модель відрізняється від конструкції тим, що вона формується шляхом виключення несуттєвих деталей. Наприклад, трьохмірний об’єкт невеликої товщини може бути представлений у вигляді двомірної оболонки. Модель створюється або в інтерактивному режимі, або автоматично. Готова абстрактна модель розбивається на кінцеві елементи, що утворюють аналітичну модель. [Програмні засоби](http://ua-referat.com/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D1%96_%D0%B7%D0%B0%D1%81%D0%BE%D0%B1%D0%B8), що дозволяють конструювати абстрактну модель і розбивати її на кінцеві елементи, називаються *препроцесорами (preprocessors).*Проаналізувавши кожний елемент, комп’ютер збирає результати разом і представляє їх в візуальному форматі. Наприклад, області з високими напругами можуть бути виділені червоним кольором. Програмні засоби, що забезпечують візуалізацію, називаються *постпроцесорами (postprocessors).*

Перевагами методів аналізу і оптимізації конструкцій полягає в тому, що вони дозволяють конструктору побачити поведінку кінцевого продукту і виявити можливі помилки до створення і [тестування](http://ua-referat.com/%D0%A2%D0%B5%D1%81%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) реальних прототипів. Так як вартість конструювання на останніх стадіях розробки і виробництва продукту зростає, рання оптимізація і удосконалення, які можливі завдяки аналітичним засобам САЕ, окупаються значним зниженням [термінів](http://ua-referat.com/%D0%A2%D0%B5%D1%80%D0%BC%D1%96%D0%BD%D0%B8) і вартості розробки.

Таким чином, технології CAD, САМ і САЕ полягають в автоматизації і підвищенні ефективності конкретних стадій життєвого циклу продукту. Розвиваючись незалежно, ці системи ще не до кінця реалізували потенціал інтеграції проектування і виробництва. Для рішення цієї проблеми була запропанована нова технологія, яка називається *комп’ютеризованим інтегрованим*[*виробництвом*](http://ua-referat.com/%D0%92%D0%B8%D1%80%D0%BE%D0%B1%D0%BD%D0%B8%D1%86%D1%82%D0%B2%D0%BE)*(computer-integrated manufacturing - CIM).*CІM намагається з’єднати автоматизації разом і перетворити їх в безперебійно і ефективно працюючу систему. CIM має на увазі використання комп’ютерної бази даних для більш ефективного управління всім підприємством, в тому числі [бухгалтерією](http://ua-referat.com/%D0%91%D1%83%D1%85%D0%B3%D0%B0%D0%BB%D1%82%D0%B5%D1%80%D0%B8%D1%8F), плануванням, доставкою та іншими задачами, а не тільки проектуванням і виробництвом, які охоплюються системами CAD, САМ і САЕ. CIM часто називають [філософією](http://ua-referat.com/%D0%A4%D1%96%D0%BB%D0%BE%D1%81%D0%BE%D1%84%D1%96) [бізнесу](http://ua-referat.com/%D0%91%D1%96%D0%B7%D0%BD%D0%B5%D1%81), а не комп’ютерною системою.

***Технологічна операція проектування***

Складність, високі [витрати](http://ua-referat.com/%D0%92%D0%B8%D1%82%D1%80%D0%B0%D1%82%D0%B8) і трудоємність [процеса](http://ua-referat.com/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81) проектування ЕІС на протязі всього життєвого цикла викликають необхідність, з одного боку, вибора адекватної економічному об'єкту технології проектування, з іншого боку, наявність ефективного інструмента управління процессом її використання. З однієї точки зору виникає потреба в побудові такої формалізованної моделі технології проектування, коли на її основі можно було б оцінити необхідність і можливість застосування технології проетування з обліком зформованих вимог до ЕІС і виділених ресурсів на економічному об'єкті, а в подальшому контролювати хід і результати проектування.

Технологічною [операцією](http://ua-referat.com/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D1%96%D1%8F) проектування системи обробки даних називається відносно самостійний фрагмент технологічного процесу, в якому визначені вхід, [вихід](http://ua-referat.com/%D0%92%D0%B8%D1%85%D1%96%D0%B4), перетворювач, [ресурси](http://ua-referat.com/%D0%A0%D0%B5%D1%81%D1%83%D1%80%D1%81%D0%B8) і засоби. Технологічна операція ТО описується векторною послідовністю:

ТО = (V, П, W, R, S),

де V - вектор входу по відношенню до даної ТО, який складається з [множини](http://ua-referat.com/%D0%9C%D0%BD%D0%BE%D0%B6%D0%B8%D0%BD%D0%B8) компонентів входу {Vi}, i=1, I; П - перетворювач ТО; W - вектор виходу по відношенню до ТО, який складається з множини компонентів виходу { Wj}, j=1,J; R - [ресурси](http://ua-referat.com/%D0%A0%D0%B5%D1%81%D1%83%D1%80%D1%81%D0%B8), які необхідні для виконання ТО; S - засоби проектування, які використовуються при виконанні ТО.

В якості компонентів входу і виходу ТО проектування можуть бути множина документів D, універсальних множин (універсумів) U, параметрів P і програм G. Графічна інтерпретація технологічної операції представлена на рис.1.



Технологічні операції графічно представляються в вигляді блоків-прямокутників, всередині яких є найменування технологічних операцій, перелік засобів проектування, що використовується і посилання на ресурси, які використовуються. Входи і виходи ТО представляються ідентифікаторами всередині кружків, від яких і до яких йдуть стрілки, які вказують вхідні і вихідні потоки.

Документ D - це описувач множини взаємопов'язаних фактів. За допомогою документів описуються об'єкти [матеріальних](http://ua-referat.com/%D0%9C%D0%B0%D1%82%D0%B5%D1%80%D1%96%D0%B0%D0%BB%D0%B8) і [інформаційних](http://ua-referat.com/%D0%86%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D1%96%D1%8F) потоків, [організаційної](http://ua-referat.com/%D0%9E%D1%80%D0%B3%D0%B0%D0%BD%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%8F) структури, технічних засобів, які необхідні для проектування і втілення ЕІС. Документи визначають або вихідні данні проектування, або кінцеві результати проектування для реалізації нової інформаційної системи, або проміжного результата, який використовується тимчасово для виконанння наступних ТО. Кінцеві документи одночасно можуть бути і проміжними. Кінцеві результати повинні бути оформленні у відповідності зі [стандартами](http://ua-referat.com/%D0%A1%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82) представлення проектної документації.

Параметр P - це характеристика, умова або деяке обмеження на систему, що проектується, які задані в явному вигляді, наприклад об'єм фінансування, строк розробки, форма представлення і т.д. Параметри можеть розглядатися як підклас документів

Програма G - це деяке проектне рішення по реалізації заданої [функції](http://ua-referat.com/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D1%96%D1%97) управлінняобєктом або по обробці даних, які записані у вигляді [функціональних](http://ua-referat.com/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D1%96%D0%BE%D0%BD%D0%B0%D0%BB%D1%96%D0%B7%D0%BC) специфікацій, програмних специфікацій, схеми алгоритму.

Універсум U - це повний перелік можливих значень деякого компоненту ТО або повний обєм знань про нього. За допомогою універсума описується множина альтернатив, [вибір](http://ua-referat.com/%D0%92%D0%B8%D0%B1%D1%96%D1%80) з якої конкретного екземпляра визначає [характер](http://ua-referat.com/%D0%A5%D0%B0%D1%80%D0%B0%D0%BA%D1%82%D0%B5%D1%80) наступних проектних рішень. Можно виділити 2 типи універсумів: проектні і інструментальні. В проектному універсумі зібрані компоненти, які орієнтовані на використання функціонуючої АІС, тобто ті компоненти, які являються результатом проектування. Інструментальні універсуми включають компоненти одного функціонального призначення, які використовуються в [процесі](http://ua-referat.com/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81) проектування.

Перетворювач П - це деяка методика або формалізований алгоритм, або машиний алгоритм [перетворення](http://ua-referat.com/%D0%9F%D0%B5%D1%80%D0%B5%D1%82%D0%B2%D0%BE%D1%80%D0%B5%D0%BD%D0%BD%D1%8F) входа технологічної операції в її вихід. [Відповідно](http://ua-referat.com/%D0%92%D1%96%D0%B4%D0%BF%D0%BE%D0%B2%D1%96%D0%B4%D1%8C) використовується ручні, автоматизовані і [автоматичні](http://ua-referat.com/%D0%90%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0) методи реалізації перетворювачів. Для формалізації перетворювачів використовуються [математичні моделі](http://ua-referat.com/%D0%9C%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D1%87%D0%BD%D1%96_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%96), блок-схеми, псевдокоди.

Ресурси R - набір людських, комп'ютерних, часових і фінансових засобів, які дозволяють виконати технологічну операцію. При чому проектувальники можуть бути [спеціалістами](http://ua-referat.com/%D0%A1%D0%BF%D0%B5%D1%86%D1%96%D0%B0%D0%BB%D1%96%D1%81%D1%82) різної кваліфікації. Наявність тих або інших ресурсів впливають на характер і технології проектування, що використовується. Наприклад, виділення мережевих комп'ютерних ресурсів, які дозволяють здійснювати [колективну](http://ua-referat.com/%D0%9A%D0%BE%D0%BB%D0%B5%D0%BA%D1%82%D0%B8%D0%B2) розробку ЕІС різними групами проектувальників і розподіленням виконаних технологічних операцій. Можко виділити 3 типи перетворювачів: ручний, людинно-машинний і машинний. До ручних відносяться - методики проведення проектних робіт. На мал.2 зображена технологічна операція з ручним перетворювачем по створенню технічного завдання ТЗ, яке включає вимоги до [функціональної](http://ua-referat.com/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D1%96%D0%BE%D0%BD%D0%B0%D0%BB%D1%96%D0%B7%D0%BC) частини, системні рішення, [економічну](http://ua-referat.com/%D0%95%D0%BA%D0%BE%D0%BD%D0%BE%D0%BC%D1%96%D0%BA%D0%B0) ефективність, строки закінчення проекту. Прикладом людинно-машинного перетворювача може біти [сценарій](http://ua-referat.com/%D0%A1%D1%86%D0%B5%D0%BD%D0%B0%D1%80%D1%96%D0%B9) інтерактивної відладки програм за допомогою інструментальних засобів.

Тема 2.3. Системи та технології управління проектуванням та життєвим циклом виробів CALS – технології.

Основні положення та принципи CALS. Задачі, що вирішуються за допомогою CALS - технологій. Склад проектної документації. Автоматизовані системи діловодства. Управління проектами.

1. КОНЦЕПЦИЯ CALS-ТЕХНОЛОГИЙ

**1.1 Единое информационное пространство**

Основное содержание концепции *CALS*-технологий, принципиально отличающее ее от других, составляют инвариантные понятия, которые реализуются (полностью или частично) в течение жизненного цикла изделия (ЖЦИ).

Эти инвариантные понятия условно делят на три группы (рис. 1.1):

• базовые принципы *CALS*;

• базовые управленческие технологии;

• базовые технологии управления данными.

К первой группе относят:

• системную информационную поддержку ЖЦИ на основе использования интегрированной информационной среды (ИИС) или единого информационного пространства (ЕИП), которое обеспечивает минимизацию затрат в ходе ЖЦ;

• информационную интеграцию за счет стандартизации информационного описания объектов управления;

• разделение программ и данных на основе стандартизации структур

данных и интерфейсов доступа к ним, ориентация на готовые коммерческие

программно-технические решения (*Commercial Of The Shelf – COTS*), соответствующие требованиям стандартов;

• безбумажное представление информации, использование электронно-цифровой подписи;

• параллельный инжиниринг (*Concurrent Engineering*);

• непрерывное совершенствование бизнес-процессов (*Business Processes Reengineering*).

Ко второй группе относят технологии управления процессами, инвариантные по отношению к объекту (продукции):

• управление проектами и заданиями (*Project Management/Workflow Management*);

• управление ресурсами (*Manufacturing Resource Planning*);

• управление качеством (*Quality Management*);

• интегрированную логистическую поддержка (*Integrated Logistic Support*).

К третьей группе относят технологии управления данными об изделии, процессах, ресурсах и среде.\_\_



 Так как стратегия *CALS* предполагает создание ЕИП для всех участников ЖЦИ (в том числе эксплуатирующих организаций), то ЕИП должно обладать следующими свойствами [3]:

• вся информация представлена в электронном виде;

• ЕИП охватывает всю информацию, созданную об изделии;

• ЕИП является единственным источником данных об изделии (прямой

обмен данными между участниками ЖЦ исключен);

• ЕИП строится только на основе международных, государственных и

отраслевых информационных стандартов;

• для создания ЕИП используются программно-аппаратные средства,

уже имеющиеся у участников ЖЦ.

Стратегия *CALS* предусматривает двухэтапный план создания ЕИП:

• автоматизация отдельных процессов (или этапов) ЖЦИ и представле-

ние данных на них в электронном виде;

• интеграция автоматизированных процессов и относящихся к ним

данных, уже представленных в электронном виде, в рамках ЕИП.

Основными преимуществами ЕИП являются:

• обеспечение целостности данных;

• возможность организации доступа к данным географически удаленных участников ЖЦИ;

• отсутствие потерь данных при переходе между этапами ЖЦИ;

• изменения данных доступны сразу всем участникам ЖЦИ;

• повышение скорости поиска данных и доступа к ним по сравнению с

бумажной документацией;

• возможность использования различных компьютерных систем для

работы с данными.

ЕИП может быть создано для организационных структур разного уров-

ня: от отдельного подразделения до виртуального предприятия или корпора-

ции. Информацию, используемую в ходе жизненного цикла, можно условно

разделить на три класса: о продукции, о выполняемых процессах, о среде, в

которой эти процессы выполняются. На каждой стадии создается набор дан-

ных, который используется на последующих стадиях (табл. 1.1). При этом

различается и эффект, получаемый от создания ЕИП (табл. 1.2).

При реализации стратегии *CALS* должны использоваться три группы

методов, называемых *CALS*-технологиями [4]:

• технологии анализа и реинжиниринга бизнес-процессов – набор ор-

ганизационных методов реструктуризации способа функционирования пред-

приятия с целью повышения его эффективности. Эти технологии нужны для

того, чтобы корректно перейти от бумажного к электронному документообо-

роту и внедрить новые методы разработки изделия;

• технологии представления данных об изделии в электронном виде –

набор методов для представления в электронном виде данных об изделии,

относящихся к отдельным процессам ЖЦИ. Эти технологии предназначены

для автоматизации отдельных процессов ЖЦ (первый этап создания ЕИП);

• технологии интеграции данных об изделии – набор методов для инте-

грации автоматизированных процессов ЖЦ и относящихся к ним данных,

представленных в электронном виде, в рамках ЕИП. Эти технологии отно-

сятся ко второму этапу создания ЕИП.

Основными компонентами *CALS*-технологий являются [5] (рис. 1.2):

• инструментальный комплекс технических и программных средств ав-

томатизированного проектирования изделий (*CAD* – *Computer Aided Design*);

• системы автоматизации технологической подготовки производства

(*CAM* – *Computer Aided Manufacturing*);

• системы инженерного анализа (*CAE* – *Computer Aided Engineering*);

• средства реализации технологии параллельного тотального проекти-

рования в режиме группового использования данных (*Concurrent*

*Engineering*);

• система управления проектными и инженерными данными (*EDM* –

*Enterprise Data Management*);



Рис. 1.2. Этапы жизненного цикла промышленных изделий и системы их автоматизации

Унификация содержания как однозначно правильная интерпретация

данных о конкретном изделии на всех этапах его ЖЦ обеспечивается разра-

боткой онтологий приложений, закрепляемых в прикладных *CALS*-

протоколах.

Унификация перечней и наименований сущностей, атрибутов и отно-

шений в определённых предметных областях является основой для единого

электронного описания изделия в *CALS*-пространстве [6]*.*

**1.2. Виртуальное предприятие**

Развитие *CALS* обусловило появление новой организационной формы

выполнения масштабных наукоемких проектов, связанных с разработкой,

производством и эксплуатацией сложной продукции так называемого «вир-

туального предприятия» – *IVE* (*Industrial Virtual Enterprise)*. Упрощенно

функционирование виртуального предприятия можно представить в виде

схемы, представленной на рис. 1.3. Виртуальное предприятие создается по-

средством объединения на контрактной основе предприятий и организаций,

участвующих в жизненном цикле продукции и связанных общими бизнес-

процессами. Информационное взаимодействие участников виртуального

предприятия осуществляется на основе общих хранилищ данных через об-

щую корпоративную или глобальную сеть [7]. Срок жизни виртуального

предприятия определяется длительностью проекта или жизненного цикла

продукции. Задача информационного взаимодействия особенно актуальна

для временно создаваемых виртуальных предприятий, состоящих из геогра-

фически удаленных друг от друга подрядчиков, субподрядчиков, поставщи-

ков с разнородными компьютерными платформами и программными реше-

ниями*.*

Виртуальное предприятие создается путем отбора требующихся чело-

веческих, организационно-методических и технологических ресурсов пред-

приятий участников с их сетевой компьютерной интеграцией. Для его эффек-

тивной работы необходимо наладить интенсивное взаимодействие реально

имеющихся специалистов и подразделений различных предприятий в вирту-

альном пространстве с помощью новейших информационных и коммуника-

ционных технологий. Это взаимодействие призвано повысить уровень коо-

перации и координации партнеров, а в конечном итоге, конкурентоспособ-

ность производимой ими продукции и, соответственно, прибыль.

Виртуальное предприятие должно обеспечивать реализацию всего

жизненного цикла изделия. При этом портфель заказов может изменяться в

короткие сроки, так же как и состав участвующих в его выполнении органи-

заций. Поэтому организация производства виртуального предприятия требу-

ет наличия специальной инфраструктуры. Типичная инфраструктура вирту-

ального предприятия, получившая широкое распространение в зарубежной

практике, включает в себя следующие основные

составляющие: cеть *Internet/Intranet*, международный стандарт *STEP*\_\_

*(Standard for the Exchange of Product model data)* для обмена данными по мо-

делям продукции и стандарты на взаимодействие прикладных программ

*(Common Object Request Broker Architecture).* При этом прикладные програм-

мы, представленные в стандарте *CORBA*, могут использовать данные, полу-

чаемые через *Internet* в формате *STEP*. При создании общей инфраструктуры

язык *IDL*, применяемый в *CORBA*, объединяется с языком описания данных

*EXPRESS*, используемым в стандарте *STEP.* Здесь *IDL* описывает интерфейсы

работы с прикладными программами, а объектно-ориентированный язык

*EXPRESS* служит для получения нормализованных моделей данных. Для ре-

альной работы используются интерфейсы *EXPRESS-G* на *C++*, *Java* или *XML*

(*Extensible Markup Language*) [1]*.*



Рис. 1.3. Упрощенная схема функционирования виртуального предприятия

Таким образом, основой инфраструктуры является *World Wide Web*

*Server*, содержащий протоколы коммуникации для организации данных и

обеспечения доступа к ним через *Internet*. Стандарт *STEP* позволяет осущест-

влять обмен данными по различным моделям продукции, что дает возмож-

ность другим приложениям понимать семантику производственной инфор-

мации. С его помощью строятся нормализованные объектные модели, назы-

ваемые «информационными моделями» и служащие для производственных

приложений. В свою очередь, открытая спецификация *CORBA* обеспечивает

применение совместных ресурсов путем поддержки обмена сообщениями

между объектами или агентами сети (например многократно используемыми

программами, составляющими приложение «клиент-сервер» в распределен-

ной среде). *СORBA*-совместимые брокеры объектных запросов не зависят от

вида платформы и могут использоваться с различными операционными сис-

темами.

Эффективность компьютерной автоматизации предприятий определя-

ется значительными успехами информационных технологий, базирующихся

на разрабатываемых в последние годы технологиях обработки и передачи

информационных потоков: *CALS, CASE, WEB*. Для полной или частичной ав-

томатизации бизнес-процесса виртуального предприятия рекомендуется

применять программные средства информационной технологии *workflow*

(поток работ или данных), что позволит передавать документы, информацию

или задания для выполнения необходимых действий от одного участника к

другому в соответствии с набором процедурных правил.

Создание виртуальных предприятий требует проработки общей схемы

совместного функционирования и взаимодействия составных частей. Это вы-

водит на первый план вопросы проектирования, анализа и при необходимо-

сти реинжиниринга внутренних и совместных бизнес-процессов, юридиче-

ского взаимодействия и интеллектуальной собственности.

**1.3. Нормативная база**

Совместное многократное использование однажды созданной инфор-

мации обеспечивается стандартизацией. Стандарты *CALS* определяют набор

правил и регламентов, в соответствии с которыми строится взаимодействие

участников процессов ЖЦИ. Существующие стандарты *CALS* можно струк-

турировать по этапам жизненного цикла и по объекту описания [8, 19],

Основной стандарт электронного описания продукции *ISO 10303 STEP*

определяет структуру базы данных об изделии, в том числе данных

о составе изделия, вариантах конфигурации, свойствах и технических характеристиках, геометрических моделях и чертежах, иллюстрациях, данные о контрактах, изменениях и т. д. На основе стандартов семейства *STEP*:

• создание структурированного электронного хранилища конструкторских данных об изделии, интегрирующего процессы разработки и получаемые результаты в единое целое;

• подготовка лицензионной документации при продаже лицензии и передаче ее в электронном виде;

• обмен данными между предприятиями, применяющими разнородные системы автоматизированного проектирования.

Стандарт *ISO 8879 (SGML)* предназначен для создания электронных документов «страничного» типа. На основе стандартов семейства *SGML* решаются следующие задачи:

• подготовка электронной эксплуатационной и ремонтной документации;

• организация обмена электронными документами между предприятиями или с заказчиком.

Следующий блок стандартов и технологий – стандарты описания процессов. Наличие формализованного описания процессов необходимо для оптимизации всего их комплекса, от разработки до эксплуатации и утилизации.

Для решения этой задачи применяется фактический международный стандарт *IDEF/0 (FIPS 183).* Электронное описание процессов представляет собой базу данных о материальных и информационных потоках, ресурсах. На ее основе решаются следующие задачи:

• анализ и реинжиниринг бизнес-процессов;

• проектирование информационной инфраструктуры, процедур и регламентов информационного взаимодействия;

• разработка регламентов и процедур обеспечения качества продукции и создания систем для обработки данных о качестве;

• разработка систем документооборота, должностных инструкций и др.

**1.3.1. Стандарт *ISO 10303 STEP***

Стандарт *ISO 10303 STEP* (*STandard for the Exchange of Product data*)

[www.iso.ch, www.prostep.org, www.pdes.de, www.steptools.com] – один из

первых в семействе специализированных *CALS-*стандартов, является характерным примером информационного стандарта нового поколения, по образу и подобию которого строятся последующие *CALS-*стандарты.

В соответствии с названием *STEP* определяет «нейтральный» формат

представления данных об изделии в виде информационной модели. Для

обеспечения возможности единообразного описания изделий в различных

прикладных областях предполагается, что создаваемые интегрированные

информационные модели (в терминах стандарта «прикладные протоколы»)

создаются на базе типовых блоков («интегрированных ресурсов»), причем

для описания схем данных используется специально введенный язык

*Express* [19].

В соответствии с этим международным стандартом конструкторское

электронное описание изделия (рис. 1.4) [2, 3] должно содержать структуру и

вариант конфигурации изделия (изделие может иметь различные версии),

геометрические модели и чертежи, свойства и характеристики составных

частей и др.\_\_



Тема 2.4. CALS - технології, стандарти IDEF, STEP, та ГОСТ Р ИСО 10303, AISO.

Роль стандартизації, сертифікації і ліцензування в процесі інформатизації. Об' єкти стандартизації. Стандарти і методи сімейства IDEF. Стандарт ISO 10303 (STEP). Стандарт ISO 15531 (MANDATE). Стандарт ISO 8879 (SGML). Ліцензування діяльності у сфері інформатизації.

Основні поняття. Завдання державної політики в галузі індустрії інформатизації.

В останні десятиліття світ переживає перехід від "індустріального суспільства" до "Суспільству інформаційному". Відбувається кардинальна зміна способів виробництва, світогляду людей, міждержавних відносин. Рівень розвитку інформаційного простору суспільства вирішальним чином впливає на економіку, обороноздатність і політику. Від цього рівня в значній мірі залежить поведінка людей, формування суспільно-політичних рухів і соціальна стабільність.

Цілями інформатизації у всьому світі і в тому числі в Росії є повне задоволення інформаційних потреб суспільства у всіх сферах діяльності, поліпшення умов життя населення, підвищення ефективності суспільного виробництва, сприяння стабілізації соціально-політичних відносин у державі на основі впровадження засобів обчислювальної техніки і телекомунікацій.

Як відомо, під всіх розвинених країнах задоволення всіх основних соціальних та індивідуальних потреб здійснюється за рахунок розповсюдження та використання інформаційних ресурсів суспільства, забезпечення доступу до них за допомогою сучасних інформаційних технологій і розвиненою інформаційно-комунікаційної структури.

Інформаційні ресурси, інфраструктури і технології в сукупності утворюють інтегровану інформаційне середовище (ІС) суспільства. Розвинена інформаційна середа служить технологічним базисом формування єдиного інформаційного простору Росії як цілісного федеративної держави, що забезпечує включення російських регіонів у соціально-економічну, політичну і культурну життя країни, послідовне входження Росії в Європейську і глобальну інформаційну інфраструктуру. Вона є необхідною умовою гнучкого та ефективного управління життям суспільства.

Процесами формування ІС - процесами інформатизації - у Росії стали серйозно займатися з початку 90-х років. Спочатку Комітет при Президентові РФ з політики інформатизації, а в даний час в результаті ряду структурних реорганізацій Мінзв'язку Росії очолює роботи по організації цих процесів, координації дій наукових і конструкторських організацій.

Незважаючи на складності, обумовлені перехідною економікою, швидким розвитком вітчизняного ринку інформаційних, комп'ютерних і телекомунікаційних технологій, державна політика інформатизації придбала в даний час концептуальну цілісність. Створені важливі правові, організаційні та економічні умови для розвитку інформаційної та комунікаційної інфраструктури, системи розповсюдження та використання інформаційних ресурсів. Істотна увага приділяється розробці законодавства в цій області. Так, у вересні 1992 року прийнятий Закон "Про правову охорону програм для електронних машин і баз даних ", в листопаді 1994 року - Закон" Про обов'язковий примірник документів ", в лютому 1995 року - Закон" Про інформації, інформатизації та захисту інформації ", в червні 1996 року - Закон "Про участь в міжнародному інформаційному обміні". З проблем інформатизації випущено велику кількість указів Президента РФ, постанов Уряду РФ, а також керівних і організаційно-методичних матеріалів різних державних організацій. Тут нам представляється корисним зупинитися на основних елементах понятійного апарату інформатизації, введених в згаданих вище, нормативно-правових документах.

Насамперед дамо визначення власне терміну "інформатизація". У Законі "Про інформації, інформатизації та захисту інформації "це поняття визначено наступним чином:

**Інформатизація**- організаційний соціально-економічний і науково-технічний процес створення оптимальних умов для задоволення інформаційних потреб і реалізації прав громадян, органів державної влади, органів місцевого самоврядування, організацій, громадських об'єднань на основі формування і використання інформаційних ресурсів.

У цьому Федеральному законі використовується ще кілька понять:

**інформація**- відомості про осіб, предмети, факти, події, явища і процеси незалежно від форми їх подання;

**документована інформація (документ)**- зафіксована на матеріальному носії інформація з реквізитами, що дозволяють її ідентифікувати;

**інформаційні процеси**- процеси збору, обробки, накопичення, зберігання, пошуку і розповсюдження інформації;

**інформаційна система**- організаційно - упорядкована сукупність документів (Масивів документів) та інформаційних технологій, у тому числі з використанням засобів обчислювальної техніки і зв'язку, що реалізують інформаційні процеси;

**інформаційні ресурси**- окремі документи і окремі масиви документів, документи і масиви документів в інформаційних системах (бібліотеках, архівах, фондах, банках даних, інших інформаційних системах);

**інформація про громадян (Персональні дані)**- відомості про факти, події і обставинах життя громадянина, що дозволяють ідентифікувати його особу;

**конфіденційна інформація**- документована інформація, доступ до якої обмежується відповідно до законодавства Російської Федерації;

**власник інформаційних ресурсів, інформаційних систем, технологій та засобів їх забезпечення**- суб'єкт, в повному обсязі реалізує повноваження володіння, користування, розпорядження зазначеними об'єктами;

**власник інформаційних ресурсів, інформаційних систем, технологій та засобів їх забезпечення**- суб'єкт, здійснює володіння і користування зазначеними об'єктами і реалізує повноваження розпорядження в межах, встановлених законом;

**користувач (Споживач) інформації**- суб'єкт, який звертається до інформаційної системі чи посереднику за отриманням необхідної йому інформації і користується нею.

Закон "Про інформації, інформатизації та захисту інформації "визначає основні напрями державної політики у сфері інформатизації. У зв'язку з важливістю цих питань наведемо відповідні формулювання закону повністю.

**Основними напрямами державної політики у сфері інформатизації є:**

вЂў забезпечення умов для розвитку і захисту всіх форм власності на інформаційні ресурси;

вЂў формування та захист державних інформаційних ресурсів; створення і розвиток федеральних і регіональних інформаційних систем і мереж, забезпечення їх сумісності та взаємодії в єдиному інформаційному просторі Російської Федерації;

вЂў створення умов для якісного та ефективного інформаційного забезпечення громадян, органів державної влади, органів місцевого самоврядування, організацій і громадських об'єднань на основі державних інформаційних ресурсів;

вЂў забезпечення національної безпеки у сфері інформатизації, а також забезпечення реалізації прав громадян, організацій в умовах інформатизації;

вЂў сприяння формуванню ринку інформаційних ресурсів, послуг, інформаційних систем, технологій, засобів їх забезпечення;

вЂў формування та здійснення єдиної науково-технічної та промислової політики у сфері інформатизації з урахуванням сучасного світового рівня розвитку інформаційних техно-логій;

вЂў підтримка проектів і програм інформатизації;

вЂў створення і вдосконалення системи залучення інвестицій та механізму стимулювання розробки та реалізації проектів інформатизації;

вЂў розвиток законодавства в сфері інформаційних процесів, інформатизації і захисту інформації.

У "Концепції формування і розвитку єдиного інформаційного простору Росії та відповідних державних інформаційних ресурсів ", схваленої рішенням Президента РФ в 1995 році, зазначено, що наявні проблеми інформатизації Росії можна вирішити тільки шляхом формування єдиного інформаційного простору Росії. Це поняття визначено в Концепції так:

**Єдине інформаційне простір**представляє собою сукупність баз і банків даних, технологій їх ведення і використання, інформаційно-телекомунікаційних систем і мереж, що функціонують на основі єдиних принципів і за загальними правилами, що забезпечує інформаційну взаємодію організацій і громадян, а також задоволення їх інформаційних потреб.

Іншими словами **, єдиний інформаційний простір складається з наступних головних компонентів:**

вЂў інформаційні ресурси, що містять дані, відомості і знання, зафіксовані на відповідних носіях інформації;

вЂў організаційні структури, що забезпечують функціонування і розвиток єдиного інформаційного простору, зокрема збір, обробку, зберігання, розповсюдження, пошук і передачу інформації;

вЂў кошти інформаційної взаємодії громадян і організацій, що забезпечують їм доступ до інформаційних ресурсів на основі відповідних інформаційних технологій, включають програмно-технічні засоби й організаційно-нормативні документи.

Організаційні структури та засоби інформаційної взаємодії утворюють **інформаційну інфраструктуру.**

**Цілями формування і розвитку єдиного інформаційного простору Росії є:**

вЂў забезпечення прав громадян на інформацію, проголошених Конституцією Російської Федерації;

вЂў створення і підтримка необхідного для сталого розвитку суспільства рівня інформаційного потенціалу;

вЂў підвищення узгодженості рішень, прийнятих федеральними органами державної влади, органами влади суб'єктів Федерації і органами місцевого самоврядування;

вЂў підвищення рівня правосвідомості громадян шляхом надання їм вільного доступу до правових і нормативним документам, які визначають їх права, обов'язки та можливості;

вЂў надання можливості контролю з боку громадян та громадських організацій за діяльністю федеральних органів державної влади, органів влади суб'єктів Федерації та органів місцевого самоврядування;

вЂў підвищення ділової та громадської активності громадян шляхом надання рівної з державними структурами можливості;

вЂў інтеграція з світовим інформаційним простором. Розвиток інформаційної інфраструктури Росії багато в чому визначається сучасним рівнем розвитку вітчизняної**індустрії інформатизації.**

**Основними завданнями державної політики в галузі індустрії інформатизації є:**

вЂў створення вітчизняних сучасних інформаційних технологій і розвиток виробництва засобів для їх реалізації;

вЂў розвиток вітчизняного виробництва сучасних систем і засобів зв'язку, телекомунікаційних мереж;

вЂў сприяння впровадженню інформаційних технологій, що використовуються в зарубіжних інформаційних системах національного і транснаціонального масштабу;

вЂў підготовка кваліфікованих кадрів для роботи в галузі інформатизації. Розглянувши основні поняття, пов'язані з процесом інформатизації, і принципи організації цього процесу в Росії, перейдемо до короткого аналізу сучасного стану інформатизації Росії та її перспективам.

Лекція 2. Інформатизація Росії. Ринок програмних засобів

Розвиток ринку в програмних засобів в Росії. Критичні інформаційні, комп'ютерні та телекомунікаційні технології.

Багато в чому завдяки послідовній реалізації розглянутих в попередньому розділі основних принципів державної політики у сфері інформатизації показники розвитку інформаційного середовища російського суспільства виглядають гідно, хоча по ряду з них Росія суттєво поступається США та інших розвинених країнах.

Інформаційні ресурси Росії є величезним за обсягом, вартості і складності комплексом, включає кілька мільйонів баз даних, електронних інформаційних масивів бібліотечних і архівних фондів. Останнім часом швидко зростає кількість російських сайтів в Інтернеті, щорічно їх кількість подвоюється. Однак не можна не відзначити, що за показником доступності інформаційних ресурсів наша країна відстає від розвинених країн.

Сьогодні в більшості великих міст Росії ефективно функціонують провайдери - організації, що забезпечують користувачам доступ в Інтернет. Широко використовується безкоштовне (некомерційне) обслуговування користувачів. Швидко розширюється і ринок мережевої комерційної інформації (відомості про компанії, товарних ринках, ринку цінних паперів, об'єктах інвестицій). Це означає серйозний крок по шляху до інформаційної економіки.

Бурхливому розвитку процесів інформатизації та, відповідно, вітчизняних територіальних комп'ютерних мереж та інформаційних систем різного роду в першій половині 90-х років в значною мірою сприяло як прискорення розвитку інфраструктури зв'язку, так і певне насичення країни персональними комп'ютерами.

В даний час вітчизняні мережеві структури (при відставанні на 1-2 роки) розвиваються в напрямках, за якими йдуть США, Великобританія, Німеччина, Франція.