ТРЕНІНГ-КУРС З ІНФОРМАЦІЙНОГО УПРАВЛІННЯ ПІДПРИЄМСТВАМИ ТА ПРОЄКТАМИ

Самостійна робота за змістовним модулем 5

5.1. Послідовність виконання робіт при традиційному підході

Послідовність виконання робіт при традиційному підході до планування структури: поетапне (згори-донизу) розбиття програми на модулі з використанням, як правило, функціонального принципу розбиття; визначення інформаційних зв'язків між модулями (тип звернення, склад даних, що передаються та повертаються); проектування окремих модулів (правила модульного програмування не накладають обмежень на внутрішню організацію окремого модуля; структурне програмування – більш суворі правила стосовно структури модуля); поетапний (знизу-вгору) процес кодування та налагодження.

Недоліки традиційного підходу, які приводять до зростання вартості розробки: складність правильного планування багаторівневої структури програми, тому що відсутня формальна методика виділення модулів; складність організації міжмодульної взаємодії, тому що помилки, яких припустилися при плануванні структури, виявляються під час проектування (а іноді навіть тільки під час тестування) модулів найвищих рівнів, коли розробку більшої частини всієї системи практично завершено; складність процесу налагодження, оскільки автономне налагодження окремого модуля потребує створення технологічного програмного оточення; необхідність проведення комплексного налагодження; використання техніки логічних схем.

Для стримання впливу цих недоліків необхідно дотримуватись технологічних заходів, що забезпечують функціональну незалежність модулів складних програмних виробів.

Спадне проектування починається з чіткої постановки задачі, яку необхідно розв'язати, і являє собою поетапне (згори-донизу) розгортання алгоритму до рівня операторів відповідної мови програмування. Уся робота в разі застосування цього методу поділяється на дві частини: планування програми, яке завершується розробкою відповідної документації, що описує ієрархічні функції програми; кодування та налагодження програми.

Мета спадного проектування: дати можливість проектувальнику оперувати на кожному кроці проектування невеликою кількістю інформації; дати засоби розбиття великої задачі на менші підзадачі, кожна з яких розглядається незалежно; забезпечити хорошу структуру програми, тобто таку структуру, яка б ще до початку кодування зробила б зрозумілими загальні та часткові функції цієї програми.

Всі ці задачі вирішуються на основі принципів структурного програмування, а саме: будь-яка програма може бути побудована з

використанням тільки трьох основних структур; будь-яка програма, яка складається тільки з базових структур, піддається послідовному перетворенню на єдиний функціональний блок.

За спадного проектування ставиться обернена задача: виходячи з єдиного функціонального елемента (загальної задачі), виконати його перетворення в складну структуру основних елементів поступовим розгортанням. Засобом такого розгортання є метод покрокового проектування (інша назва – метод послідовних наближень).

Етапи проектування при використанні методу послідовних наближень:

- проаналізувати умови задачі і описати згідно з установленими правилами вихідну інформацію у вигляді ієрархічної структури;
- проаналізувати умови задачі і описати згідно з установленими правилами вхідні дані у вигляді ієрархічної структури;
- визначити методи і способи отримання вихідної інформації із заданих вхідних даних;
- виконати покрокове проектування алгоритму;
- визначити модульну структуру програми (спробувати вкласти фрагменти алгоритму таким чином, щоб були дотримані правила модульності);
- виконати за встановленими правилами опис алгоритму хибного модуля;
- підготувати зовнішні специфікації кожного модуля.

Суть покрокової деталізації алгоритму: проектування виконується дрібними кроками, на кожному з яких приймаються алгоритмічні рішення з обробки тільки одного рівня ієрархії даних.

Умови успішного застосування спадного проектування: використання на кожному рівні невеликого обсягу даних та відвертання від несуттєвих для даного рівня деталей задачі, а саме: на кожному кроці проектування користуйтеся тільки одним рівнем ієрархії структури даних, що обробляються; ретельно проектуйте структуру даних, що обробляються; дотримуйтесь вимоги: перехід до проектування модулів наступного рівня можливий тільки тоді, коли модулі чергового рівня описані згідно з установленими правилами і ретельно перевірені; забезпечення наочного сприйняття опису вхідних даних кожного рівня, його функцій і даних, що сформовані в результаті їх виконання.

5.2. Мова проектування програм

Недоліки техніко-логічних схем: розробнику доводиться мислити категоріями конкретної мови програмування, а іноді й категоріями нижчого рівня, а не змістом задачі, що розглядається; за допомогою логічних схем структура галуження програми проглядається достатньо чітко тільки в тому випадку, коли вся схема розміщена на одному аркуші (у разі розміщення її на кількох аркушах уявлення про структуру губиться).

Вимоги до мови проектування: простота вивчення та використання; можливість опису алгоритму у зручній для людини формі, що спрощує процес алгоритмізації та полегшує досягнення потрібної якості програм; можливість формалізації алгоритму в термінах абстрактних об'єктів (даних, операторів, умов, структур і т. ін.), а також у термінах операцій з об'єктами, властивості яких добре вивчені; багаторівневість подання структури даних для фіксації взаємозв'язаних процесів під час розробки алгоритмів; можливість подання алгоритму, з одного боку, у формі, зручній для людини, тобто природною мовою, а з іншого – у формалізованому вигляді; незалежність від мови програмування, тобто можливість перекладу на будь-яку мову програмування для забезпечення адекватності алгоритмів та асоційованих з ними програм.

Загальна характеристика псевдокоду, його правила та переваги

Основне призначення мови проектування – зниження трудомісткості розробки алгоритмів, написання програм, їх налагодження, подальшої модифікації з одночасним підвищенням якості програм, що створюються. Тобто, при проектуванні програм потрібна певна система запису, яка, з одного боку, дозволяє розробникові концентрувати увагу на змісті проблеми, що вирішується, а з іншого – зручна для наступної стадії перекладу алгоритму на мову програмування.

Мета мови проектування: створити передумови для покращання наочності і зрозумілості фрагментів алгоритму з метою спрощення як самого процесу проектування, так і подальшого кодування; створити передумови для підвищення наочності та зрозумілості тексту програми з метою спрощення її налагодження та подальшої модифікації.

Мова проектування програм – це частково формалізований спосіб запису для наочного текстового подання схем алгоритмів і програм, що розробляються відповідно до правил структурного програмування.

Структура мови проектування: заданий набір операторів, побудованих за принципом мов програмування і призначений для опису логіки задачі; неформальна частина, що містить поняття, які найбільш відповідають змістовному опису задачі.

Правила псевдокоду:

1. Текст псевдокоду містить функції та керуючі інструкції.

2. Як функції припускається використовувати назву функцій, що являють собою речення природною мовою, яка відбиває зміст дії; будь-які слушні формальні позначення (математичні позначення, якщо вони зрозумілі без пояснень; поширені оператори мови програмування); комбінації формалізованого та неформалізованого запису.

3. Як керуючі інструкції допускається використовувати альтернативні інструкції, інструкцію вибору, циклічну інструкцію.

Суть спадного кодування: по завершенню проектування верхнього рівня програми і до початку проектування наступних рівнів мають бути написані коди верхнього рівня, а потім проектуються і кодуються модулі другого рівня і т. д.

Переваги: у процесі кодування можуть бути виявлені ті чи інші недоліки проектування тільки тому, що під час кодування розробник вимушений старанно продумувати свої дії; спадне кодування дозволяє раніше перейти до спадного тестування; паралельність виконання етапів дозволяє скоротити загальні строки розробки.

Суть спадного тестування: спочатку тестується модуль найвищого рівня і залежно від складності задачі модулі одного чи двох більш низьких рівнів. Після того, як це логічне ядро програми випробуване настільки, що з'являється впевненість у правильності реалізації основних інтерфейсів, приєднується решта рівнів логічної структури програми. Коли система поповнюється найнижчим рівнем модулів, задача виявляється розв'язаною, і відпадає необхідність у комплексному налагодженні.

Методологія спадного тестування передбачає використання фіктивних модулів (модулів заглушок). Це означає, що коли модулі вищих рівнів звертаються до модулів більш низьких рівнів, а ті ще не готові, то ці модулі нижчих рівнів можуть бути подані одним з варіантів: пряма передача управління на вихід з модуля, якщо функція, що має виконуватися, не є критичною (порожня процедура); друкування повідомлення, яке вказує на те, що управління передано на вихід з модуля; видача постійного значення вихідних даних; видача випадкового значення вихідних даних; реалізація спрощеного варіанта версії модуля.

Переваги спадного тестування: дозволяє на ранньому етапі розробки визначити основні недоліки майбутньої системи; відпадає потреба у комплексному тестуванні.