УДК 004.415.2

Л.В. Передерій, Бердянський університет менеджменту і бізнесу

СИСТЕМНЕ ПРОЕКТУВАННЯ ІНФОРМАЦІЙНИХ СИСТЕМ

Передерій Л.В.
Системне проектування інформаційних систем
Розглядається один з можливих методів розробки інформаційних систем, який базується на моделі життєвого циклу «водоспад», детально досліджені роботи на кожному з етапів моделі, особлива увага приділена етапу проектування, як найбільш вагомому у цьому методі. Метод може бути рекомендований для розробки проектів систем певного класу, під час розробки яких можливі сильні зміни вимог до проекту.
Ключові слова: системне проектування, модель, життєвий цикл, проект.

Передерий Л.В.
Системное проектирование информационных систем
Рассматривается один из возможных методов разработки информационных систем, который базируется на модели жизненного цикла «водопад», детально исследованы работы на каждом из этапов модели, особенное внимание уделено этапу проектирования, как наиболее весомому в этом методе. Метод может быть рекомендован для разработки проектов систем определенного класса, во время разработки которых возможны сильные изменения требований к проекту.
Ключевые слова: системное проектирование, модель, жизненный цикл, проект.

На даний час складність інформаційних систем постійно зростає, кількість різних методів і методологій їх розробки велика, і правильно обрати оптимальний метод у кожному конкретному випадку досить складно. Дослідження існуючих методологій проектування інформаційних систем (ІС) показує узагальнений підхід до різного класу систем з використанням однієї з класичних моделей життєвого циклу програмного забезпечення. Життєвим циклом програмного забезпечення є модель його створення і використання. Модель відображає його різні стани, починаючи з моменту виникнення необхідності в даному програмному забезпеченні (ПЗ) і закінчуючи моментом

його повного виходу з вживання у всіх користувачів. Відомі наступні моделі життєвого циклу: каскадна, поетапна і спіральна [1, с. 11].

Каскадна модель припускає перехід на наступний етап тільки після повного завершення робіт на попередньому етапі. Поетапна модель з проміжним контролем забезпечує розробку ПЗ ітераціями з циклами зворотного зв'язку між етапами. Міжетапні коректування дозволяють зменшити трудомісткість процесу розробки в порівнянні з каскадною моделлю; час життя кожного з етапів розтягується на весь період розробки. Спіральна модель приділяє особливу увагу початковим етапам розробки - виробленню стратегії, аналізу і проектуванню, де реалізуємість тих або інших технічних рішень перевіряється і обгрунтовується за допомогою створення прототипів (макетування). Кожен виток спіралі припускає створення якоїсь версії продукту або якого-небудь його компоненту, при цьому уточнюються характеристики і цілі проекту, визначається його якість і плануються роботи наступного витка спіралі [2, с. 20]. Кожна з класичних моделей використовується з огляду на специфіку розроблювальної системи.

Ціллю статті є дослідження оптимального способу створення систем певного класу, а саме:

- загальноуправлінські ІС;

- спеціалізовані ІС по галузях виробництва, наприклад, банківські облікові і управлінські системи;

- спеціалізовані ІС по видах діяльності, наприклад, управління роботою складу, система маркетингових досліджень, аналітична система для роботи на фондовому ринку і ін.;

- адаптивні універсальні ІС по вживаних методах обробки інформації, наприклад, електронний архів, корпоративна система управління процесом виконання офісних робіт, система статистичних розрахунків і ін.

У якості моделі життєвого циклу пропонується схема «водоспад» (рис. 1), яка ефективно підходить для проектів, замовники яких дуже часто змінюють

вимоги до проекту майбутньої системи, що реально відбувається при створенні вищеозначених систем.

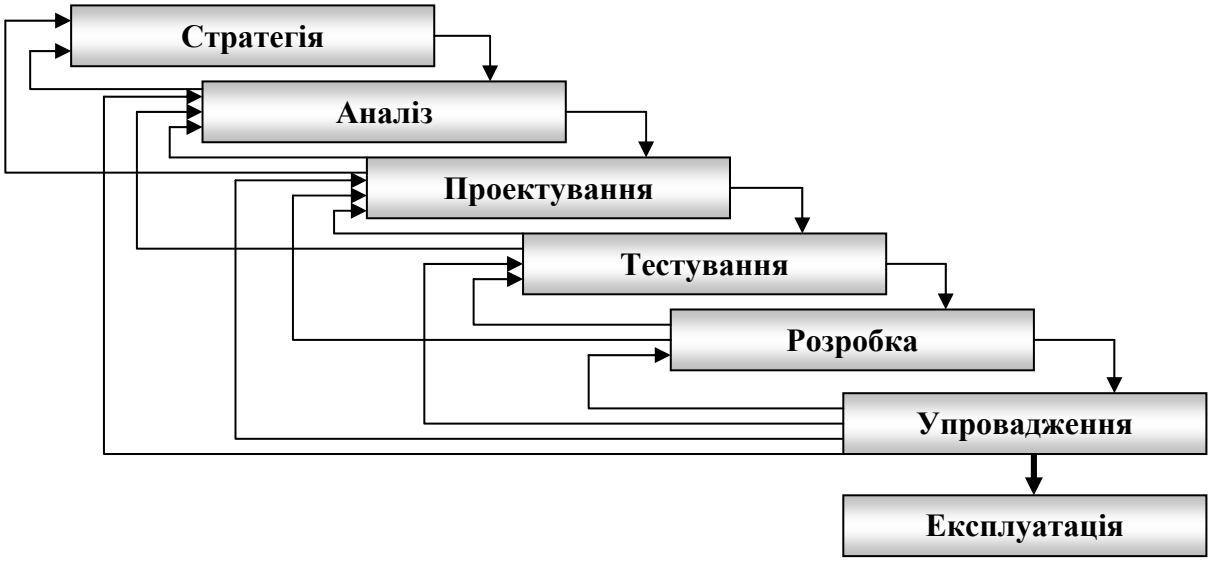

Рис. 1. Схема "водоспаду"

Розглянемо кожен з етапів, докладніше зупинившись на етапі проектування.

Визначення стратегії припускає обстеження системи. Основне завдання обстеження - оцінка реального об'єму проекту, його цілей і завдань, а також отримання визначень суті і функцій на високому рівні. На цьому етапі притягуються висококваліфіковані бізнес-аналітики, які мають постійний доступ до керівництва фірми; етап припускає тісну взаємодію з основними користувачами системи і бізнес-експертами. Основне завдання взаємодії - отримати якомога повнішу інформацію про систему (повне і однозначне розуміння вимог замовника) і передати дану інформацію у формалізованому вигляді системним аналітикам для подальшого проведення етапу аналізу. Як правило, інформація про систему може бути отримана в результаті бесід або семінарів з керівництвом, експертами і користувачами. Таким чином визначаються суть даного бізнесу, перспективи його розвитку і вимоги до системи.

Після закінчення основної стадії обстеження системи технічні фахівці формують вірогідні технічні підходи і приблизно розраховують витрати на

апаратне забезпечення, програмне забезпечення, що купується, і розробку нового програмного забезпечення (що, власне, і передбачається проектом).

Результатом етапу визначення стратегії є документ, де чітко сформульовано, що отримає замовник, якщо погодиться фінансувати проект; коли він отримає готовий продукт (графік виконання робіт); скільки це коштуватиме (для крупних проектів повинен бути складений графік фінансування на різних етапах робіт). У документі повинні бути відбиті не тільки витрати, але і вигода, наприклад час окупності проекту, очікуваний економічний ефект (якщо його вдається оцінити).

У документі обов'язково повинні бути описані:

- обмеження, ризики, критичні чинники, що впливають на успішність проекту, наприклад час реакції системи на запит є заданим обмеженням, а не бажаним чинником;

- сукупність умов, при яких передбачається експлуатувати майбутню систему: архітектура системи, апаратні і програмні ресурси, що надаються системі, зовнішні умови її функціонування, склад людей і робіт, які забезпечують безперебійне функціонування системи;

- терміни завершення окремих етапів, форма здачі робіт, ресурси, що привертаються в процесі розробки проекту, міри по захисту інформації;

- опис виконуваних системою функцій;

- майбутні вимоги до системи у разі її розвитку, наприклад можливість роботи користувача з системою за допомогою Інтернету і т.п.;

- суть, необхідна для виконання функцій системи;

- інтерфейси і розподіл функцій між людиною і системою;

- вимоги до програмних і інформаційних компонентів ПЗ, вимоги до СУБД (якщо проект передбачається реалізовувати для декількох СУБД, то вимоги до кожної з них, або загальні вимоги до абстрактної СУБД і список СУБД, що рекомендуються для даного проекту, які задовольняють заданим умовам);

- опис того, що не буде реалізоване в рамках проекту.

Виконана на даному етапі робота дозволяє відповісти на питання, чи варто продовжувати даний проект і які вимоги замовника можуть бути задоволені за тих або інших умов. Може так статися, що проект продовжувати не має сенсу, наприклад через те, що ті або інші вимоги не можуть бути задоволені за якимись об'єктивними причинами. Якщо ухвалюється рішення про продовження проекту, то для проведення наступного етапу аналізу вже є уявлення про об'єм проекту і кошторис витрат.

Етап аналізу припускає докладне дослідження бізнес-процесів (функцій, визначених на етапі вибору стратегії) і інформації, необхідної для їх виконання (сутності, їх атрибутів і зв'язків (відносин)). На цьому етапі створюється інформаційна модель, а на наступному за ним етапі проектування - модель даних [3, с. 46].

Вся інформація про систему, зібрана на етапі визначення стратегії, формалізується і уточнюється на етапі аналізу. Особливу увагу слід приділити повноті переданої інформації, аналізу інформації на предмет відсутності суперечностей, а також пошуку невживаною взагалі або інформації, що дублюється. Як правило, замовник не відразу формує вимоги до системи в цілому, а формулює вимоги до окремих її компонентів. Тому необхідно приділити увагу узгодженості цих компонентів. Аналітики збирають і фіксують інформацію в двох взаємозв'язаних формах:

1) функції - інформація про події і процеси, які відбуваються в бізнесі;

2) сутності - інформація про речі, що мають значення для організації і про яких щось відоме.

Двома класичними результатами аналізу є:

1) ієрархія функцій, яка розбиває процес обробки на складові частини (що робиться і з чого це складається);

2) модель "сутність-зв'язок (Entry Relationship model, ER-модель) ", яка описує сутності, їх атрибути і зв'язки (відносини) між ними.

Ці результати є необхідними, але не достатніми. До достатніх результатів слід віднести діаграми потоків даних і діаграми життєвих циклів сутності.

Досить часто помилки аналізу виникають при спробі показати життєвий цикл сутності на діаграмі ER [4, с. 23]. Розглянемо три найбільш часто вживані методології структурного аналізу:

1) діаграми "сутність-зв'язок (Entity-Relationship Diagrams, ERD) ", які служать для формалізації інформації про сутності і їх відносини;

2) діаграми потоків даних (Data Flow Diagrams, DFD), які служать для формалізації представлення функцій системи;

3) діаграми переходів станів (State Transition Diagrams, STD), які відображають поведінку системи, залежну від часу; діаграми життєвих циклів сутності відносяться саме до цього класу діаграм.

ER-діаграми (рис. 2) використовуються для розробки даних і є стандартним способом визначення даних і відносин між ними. Таким чином, здійснюється деталізація сховищ даних.

ER-діаграма містить інформацію про сутності системи і способи їх взаємодії, включає ідентифікацію об'єктів, важливих для наочної області (сутності), властивостей цих об'єктів (атрибутів) і їх відносин з іншими об'єктами (зв'язків). У багатьох випадках інформаційна модель дуже складна і містить безліч об'єктів.

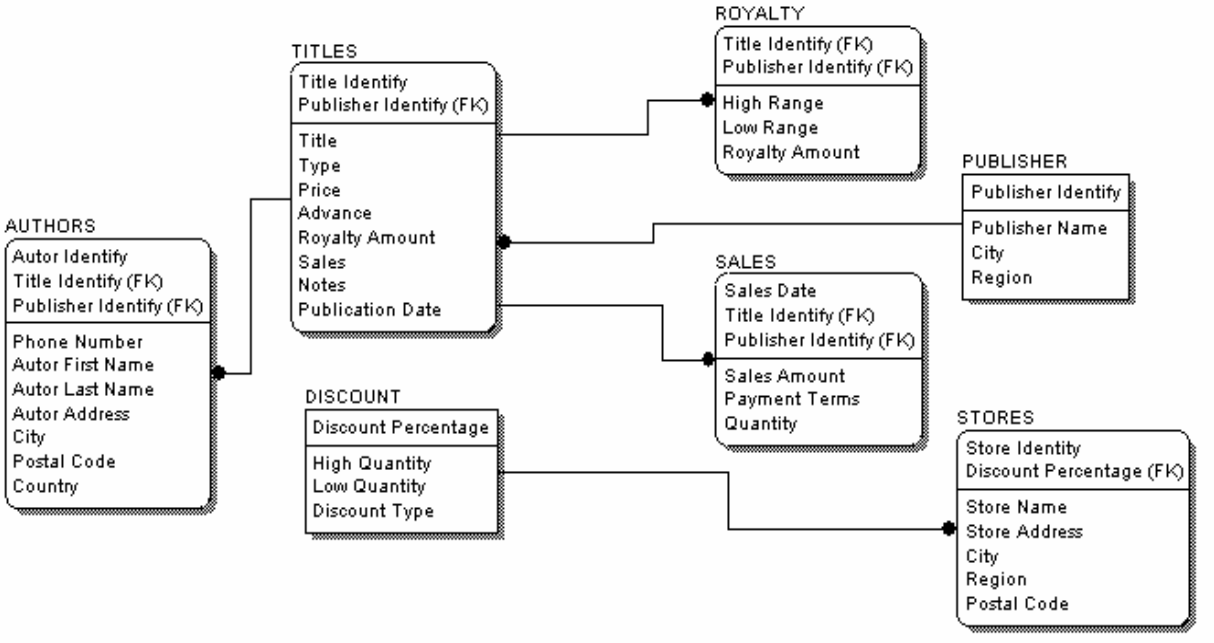

Рис. 2. Приклад ER-діаграми

Логічна DFD (рис. 3) показує зовнішні по відношенню до системи джерела і адресати даних, ідентифікує логічні функції (процеси) і групи елементів даних, що пов'язують одну функцію з іншою (потоки), а також ідентифікує сховища (накопичувачі) даних, до яких здійснюється доступ.

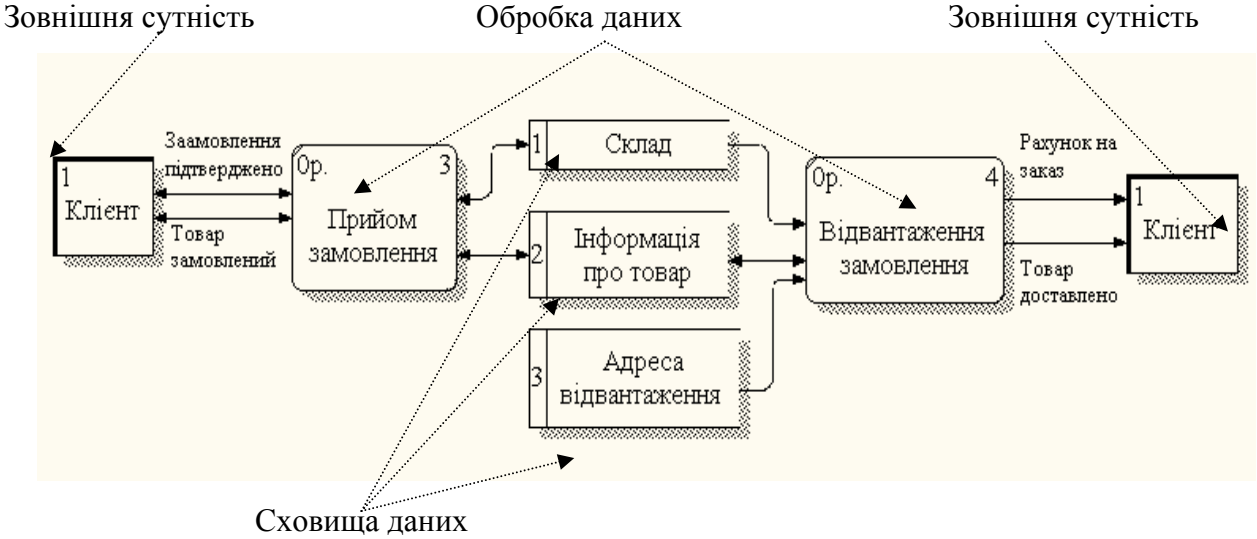

Рис. 3. Приклад DFD

Структури потоків даних і визначення їх компонентів зберігаються і аналізуються в словнику даних. Кожна логічна функція (процес) може бути деталізована за допомогою DFD нижнього рівня; коли подальша деталізація перестає бути корисною, переходять до виразу логіки функції за допомогою специфікації процесу. Вміст кожного сховища також зберігають в словнику даних, модель даних сховища розкривається за допомогою ER-діаграм.

Зокрема, в DFD не показуються процеси, які управляють власне потоком даних і не приводяться відмінності між допустимими і неприпустимими шляхами. DFD містять безліч корисної інформації, а, крім того:

- дозволяють представити систему даних;

- ілюструють зовнішні механізми подачі даних, які зажадають наявність спеціальних інтерфейсів;

- дозволяють представити як автоматизовані, так і ручні процеси системи;

- виконують орієнтоване на дані секціонування всієї системи.

Життєвий цикл сутності відноситься до класу STD-діаграм (рис. 4).

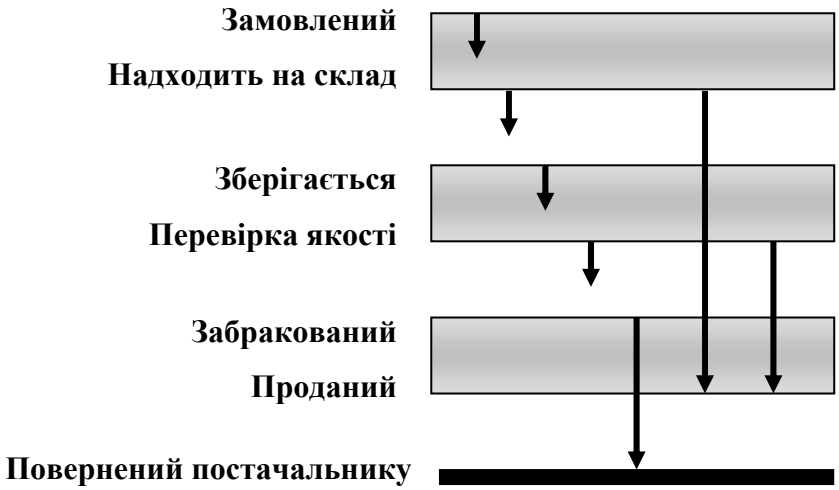

Рис. 4. Приклад діаграми життєвого циклу

Ця діаграма відбиває зміну стану об'єкту з часом. Наприклад, розглянемо перебування товару на складі: товар може бути замовлений у постачальника, поступити на склад, зберігатися на складі, проходити контроль якості, може бути проданий, забракований, повернений постачальникові. Стрілки на діаграмі показують допустимі зміни станів. Існує декілька різних варіантів зображення подібних діаграм, на рисунку приведений лише один з них.

На етапі аналізу відбувається уточнення вибраних для кінцевої реалізації апаратних і програмних засобів. Для цього можуть притягуватися групи тестування, технічні фахівці. При проектуванні інформаційної системи важливо врахувати і подальший розвиток системи, наприклад зростання об'ємів оброблюваних даних, збільшення інтенсивності потоку запитів, зміна вимог надійності інформаційної системи.

На етапі аналізу визначаються набори моделей завдань для отримання порівняльних характеристик тих або інших СУБД, які розглядалися на етапі визначення стратегії для реалізації інформаційної системи. На етапі визначення стратегії може бути здійснений вибір однієї СУБД. Даних про систему на етапі аналізу вже набагато більше, і вони докладніші. Отримані дані, а також характеристики, передані групами тестування, можуть показати, що вибір СУБД на етапі визначення стратегії був невірним і що вибрана СУБД не може

задовольняти тим або іншим вимогам інформаційної системи. Такі ж дані можуть бути отримані щодо вибору апаратної платформи і операційної системи. Отримання подібних результатів ініціює зміна даних, отриманих на етапі визначення стратегії, наприклад перераховується кошторис витрат на проект.

Вибирання засобів розробки також уточнюється на етапі аналізу. Внаслідок того, що етап аналізу дає повніше уявлення про інформаційну систему, ніж воно було на етапі визначення стратегії, план робіт може бути скоректований. Якщо вибраний на попередньому етапі засіб розробки не дозволяє виконати ту або іншу частину робіт в заданий термін, то ухвалюється рішення про зміну термінів (як правило, це збільшення терміну розробки) або про зміну засобу розробки. Здійснюючи вибирання тих або інших засобів, слід враховувати наявність висококваліфікованого персоналу, який володіє вибраними засобами розробки, а також наявність адміністраторів вибраної СУБД. Ці рекомендації також уточнюватимуть дані етапу вибору стратегії (сукупність умов, при яких передбачається експлуатувати майбутню систему).

Уточнюються також обмеження, ризики, критичні чинники. Якщо які-небудь вимоги не можуть бути задоволені в інформаційній системі, реалізованій з використанням СУБД і програмних засобів, вибраних на етапі визначення стратегії, то це також ініціює уточнення і зміну отримуваних даних (зрештою кошториси витрат і планів робіт, а можливо, і зміна вимог замовника до системи, наприклад їх ослаблення). Детальніше описуються ті можливості, які не будуть реалізовані в системі.

На етапі аналізу притягуються групи тестування, наприклад для отримання порівняльних характеристик передбачуваних до використання апаратних платформ, операційних систем, СУБД, іншого оточення. Крім того, на даному етапі визначається план робіт по забезпеченню надійності інформаційної системи і її тестування. Для будь-яких проектів доцільним є залучення тестерів на ранніх етапах розробки, зокрема на етапі аналізу і проектування. Якщо проектне рішення виявилося невдалим і це виявлено надто

пізно - на етапі розробки або, що ще гірше, на етапі впровадження в експлуатацію, - то виправлення помилки проектування може обійтися дуже дорого. Чим раніше групи тестування виявляють помилки в інформаційній системі, тим нижче вартість супроводу системи. Час на тестування системи і на виправлення виявлених помилок слід передбачати не тільки на етапі розробки, але і на етапі проектування.

Системне проектування розглядається як набір методів і організаційна дисципліна, призначені для проектування інформаційних систем певних видів. Проектування інформаційних систем завжди починається з визначення мети проекту. Основне завдання будь-якого успішного проекту полягає в тому, щоб на момент запуску системи і протягом всього часу її експлуатації можна було забезпечити:

- необхідну функціональність системи і ступінь адаптації до умов її функціонування, які можуть змінюватися;

- необхідну пропускну спроможність системи;

- необхідний час реакції системи на запит;

- безвідмовну роботу системи в необхідному режимі, іншими словами - готовність і доступність системи для обробки запитів користувачів;

- простоту експлуатації і підтримки системи;

- необхідну безпеку.

Продуктивність є головним чинником, що визначає ефективність системи. Хороше проектне рішення служить основою високопродуктивної системи. Проектування інформаційних систем охоплює три основні області:

- проектування об'єктів даних, які будуть реалізовані в базі даних;

- проектування програм, екранних форм, звітів, які забезпечуватимуть виконання запитів до даних;

- урахування конкретного середовища або технології, а саме: топології мережі, конфігурації апаратних засобів, використовуваної архітектури (файл-сервер або клієнт-сервер), паралельної обробки, розподіленої обробки даних і тому подібне.

Дуже часто проектування описують як окремий етап розробки проекту між аналізом і розробкою. Проте насправді чіткого ділення етапів розробки проекту немає - проектування, як правило, не має явно вираженого початку і закінчення і часто продовжується на етапах тестування і реалізації. І етап аналізу, і етап проектування містять елементи роботи тестерів, наприклад, для отримання експериментального обгрунтування вибору того або іншого рішення, а також для оцінки критеріїв якості отримуваної системи. На етапі експлуатації мається на увазі також і супровід системи.

На етапі проектування формується модель даних. Проектувальники у якості початкової інформації отримують результати аналізу. Кінцевим продуктом етапу проектування є:

- схема бази даних (на підставі ER-моделі, розробленої на етапі аналізу);
- набір специфікацій модулів системи (вони будуються на базі моделей функцій).

Якщо проект невеликий, то як аналітики, проектувальники і розробники можуть виступати одні і ті ж люди. Виникає питання: наскільки взагалі актуальна передача результатів самому собі? Як правило, актуальна. Часто це допомагає, наприклад, знайти не описані взагалі, нечітко описані, суперечливо описані компоненти системи.

Всі специфікації повинні бути точними. План тестування системи допрацьовується також на цьому етапі розробки. У багатьох проектах результати етапу проектування оформляються єдиним документом, який називають технічною специфікацією. У нім також описують прийнятий підхід до вирішення яких-небудь складних технічних питань.

При проектуванні виникає необхідність реєструвати всі обговорювані варіанти і остаточні рішення. Проектувальники деколи міняють первинні рішення. Це може відбуватися тому, що з часом учасники проекту забувають аргументи на користь ухваленого рішення. Подібну інформацію можна зберігати в репозитарії використовуваного CASE-засобу, в текстових файлах, просто на папері. Журнал проектування є корисним матеріалом для нових

членів груп проектувальників, а також для розробників і тестувальників [5, с. 432].

Ретельне планування важливе для будь-якого проекту. Це входить в обов'язки керівника проекту і керівника групи проектування (консультації з аналітиками в цьому випадку будуть обов'язковими). Це дозволяє:

- розбити глобальне завдання на невеликі, незалежні завдання. Такими завданнями легко управляти, такі завдання легко реалізовувати;

- визначити контрольні дати (етапи здачі), які дозволять визначити, наскільки успішно просувається проект, які напрями відстають, які недовантажені, які працюють успішно. Це дозволяє виявити відставання від термінів здачі і вчасно запобігти авралам;

- визначити залежності між завданнями, а також послідовність завершення завдань;

- прогнозувати завантаження персоналу, наймання тимчасових працівників, залучення інших груп розробників, залучення консультантів (якщо це необхідно);

- отримати чітке уявлення про те, коли можна почати етап реалізації;

- отримати чітке уявлення про те, коли можна почати етап дослідної експлуатації.

Замовники завжди хочуть, щоб план виконання робіт залишався незмінним. На практиці цього рідко вдається досягти в повному об'ємі. Певним компромісом тут може стати незмінність встановлених термінів здачі компонентів системи в експлуатацію.

Розгляд результатів аналізу - це процес передачі інформації від аналітиків проектувальникам. На практиці це інтерактивний процес. У проектувальників неминуче виникатимуть питання до аналітиків, і навпаки. Інформація про систему постійно уточнюватиметься. При розробці схеми бази даних може змінитися інформаційна модель, отримана на етапі аналізу, наприклад, тому, що наявне проектне рішення нестабільне або поволі працює при реалізації його за допомогою вибраної СУБД або через інші причини. Перевірити, чи охоплює

аналіз всі бізнес-процеси системи (тобто, здійснити перевірку на повноту), проектувальники не в змозі, але перевірку інформаційної моделі на несуперечність і коректність проектувальники провести можуть. Це дозволяє відстежити помилки в інформаційній моделі і не повторити їх в моделі даних. Якщо результати зберігаються в репозитарії CASE-засобу, то така перевірка на коректність може бути проведена автоматично.

Робота проектувальників бази даних в значній мірі залежить від якості інформаційної моделі. Інформаційна модель не повинна містити ніяких незрозумілих конструкцій, які не можна реалізувати в рамках вибраної СУБД. Слід зазначити, що інформаційна модель створюється для того, щоб на її основі можна було побудувати модель даних, тобто повинна враховувати особливості реалізації вибраної СУБД. Якщо ті або інші особливості СУБД не дозволяють відобразити в моделі даних те, що описує інформаційна модель, значить, треба міняти інформаційну модель, оскільки виробник СУБД навряд чи оперативно мінятиме власне СУБД заради нового конкретного проекту, хоча це також не виключається.

Побудова логічної і фізичної моделей даних є основною частиною проектування бази даних. Отримана в процесі аналізу інформаційна модель спочатку перетвориться в логічну, а потім у фізичну модель даних. Після цього для розробників інформаційної системи створюється пробна база даних. З нею починають працювати розробники коду. У ідеалі до моменту початку розробки модель даних повинна бути стійка. Проектування бази даних не може бути відірване від проектування модулів і додатків, оскільки бізнес-правила можуть створювати об'єкти в базі даних, наприклад серверні обмеження.

Паралельно з проектуванням схеми бази даних потрібно виконати проектування процесів, щоб отримати специфікації всіх модулів. Якщо частина бізнес-логіки зберігається в базі даних (обмеження, процедури, що зберігаються), то обидва ці процеси проектування тісно зв'язані. Головна мета проектування полягає у відображенні функцій, отриманих на етапі аналізу, в модулі інформаційної системи. Визначення модулів розкриваються в технічній

специфікації програм. Можливо, що деякі атомарні функції, отримані на етапі аналізу, взагалі не будуть відображені в які-небудь модулі, а будуть перетворені в ручні процедури або принципи роботи.

Не слід відкладати вибирання засобів розробки на найостанніший момент. Якщо проектувальник не дуже добре уявляє собі набір засобів, які будуть використані для розробки проекту, то слід спершу скласти перелік можливих засобів, потім провести консультації з технічними фахівцями, оцінити, з якими засобами персонал вже працював, а які є для них абсолютно новими. Часто вибирання засобів розробки визначає саме чинник кваліфікації персоналу.

При проектуванні модулів визначають розмітку меню, вид вікон, гарячі клавіші і пов'язані з ними виклики.

Проектування процесу тестування, як правило, слідує за процесом функціонального проектування і проектування схеми бази даних. На цьому етапі можна використовувати складні схеми тестування, а можна обмежитися і простими. Коли генерація модуля завершена, виконують автономний тест, який переслідує дві основні цілі:

1) виявлення відмов модуля (жорстких збоїв);

2) відповідність модуля специфікації (наявність всіх необхідних функцій, відсутність зайвих функцій).

Після того, як автономний тест пройшов успішно, група модулів, що згенерували, проходить тести зв'язків, які повинні відстежити взаємний вплив модулів.

Далі група модулів тестується на надійність роботи, тобто проходять, по-перше, тести імітації відмов системи, а по-друге, тести напрацювання на відмову. Перша група тестів показує, наскільки добре система відновлюється після збоїв програмного забезпечення, відмов апаратного забезпечення. Друга група тестів визначає ступінь стійкості системи при штатній роботі і дозволяє оцінити час безвідмовної роботи системи. У комплект тестів стійкості повинні входити тести, що імітують пікове навантаження на систему.

Потім весь комплект модулів проходить системний тест - тест внутрішнього приймання продукту, що показує рівень його якості. Сюди входять тести функціональності і тести надійності системи.

Останній тест інформаційної системи — приймально-здавальні випробування. Такий тест передбачає показ інформаційної системи замовникові і повинен містити групу тестів, що моделюють реальні бізнес-процеси, щоб показати відповідність реалізації вимогам замовника.

Кожна інформаційна система містить певні вимоги до захисту від несанкціонованого доступу, до реєстрації подій системи, аудиту, резервного копіювання, відновлення інформації, які на початку проектування повинні бути формалізовані аналітиками. Проектувальники будують стратегію безпеки системи. Зокрема, ними повинні бути визначені категорії користувачів системи, які мають доступ до тих або інших даних за допомогою відповідних компонентів. Крім того, визначаються об'єкти і суб'єкти захисту. Стратегія безпеки не обмежується тільки ПЗ — це повинен бути цілий комплекс заходів і правил ведення бізнесу. Потрібно чітко визначити, який рівень захисту даних необхідний для кожного з компонентів інформаційної системи, і виділити критичні дані, доступ до яких строго обмежений. Користувачі інформаційної системи реєструються, тому проектуються модулі, що відповідають за ідентифікацію і аутентифікацію користувача. У більшості СУБД реалізований регламентований доступ до об'єктів даних (наприклад, до таблиць). Якщо ж потрібне обмеження доступу власне до даних (до окремих записів в таблиці, до окремих полів запису в таблиці і тому подібне), то слід реалізувати мандатний захист. Проектувальники повинні мати чітке уявлення про те, який рівень захисту тієї або іншої одиниці інформації є необхідним, а який достатнім [6, с. 520].

Питання відновлення, зберігання резервних копій бази даних, архівів бази даних відносяться до заходів підтримки безперебійного функціонування інформаційної системи. Необхідно уважно вивчити можливості, СУБД, що

надаються, а потім проаналізувати, як слід використовувати можливості СУБД для забезпечення необхідного рівня безперебійної роботи системи.

Результати проектування відбиваються в документі — функціональній специфікації. Цей документ пишеться для замовника, щоб отримати його санкцію на завершення проектування і початок розробки, і зазвичай не містить великої кількості технічних деталей. Другий документ — технічна специфікація, що є основним документом для розробників моделей і груп тестування, у якому описані деталі проекту. Якщо використовувалися CASE-засоби, то технічна специфікація обов'язково містить ряд звітів з репозитарія.

При реалізації проекту важливо координувати групу розробників. Всі розробники повинні підкорятися жорстким правилам контролю початкових тестів. Група розробників, отримавши технічний проект, починає писати код модулів, і в цьому випадку основне завдання полягає в тому, щоб з'ясувати специфікацію. Проектувальник вказав, що необхідно зробити, а розробник визначає способи виконання.

На етапі розробки здійснюється тісна взаємодія проектувальників, розробників і груп тестерів. У разі інтенсивної розробки тестер фактично є членом групи розробки. Проектувальник на даному етапі виконує функції «ходячого довідника», оскільки постійно відповідає на питання розробників, що стосуються технічної специфікації. Найчастіше на етапі розробки міняються інтерфейси користувача. Це обумовлено у тому числі і тим, що модулі періодично демонструються замовникові. Істотно можуть мінятися і запити до даних. Взаємодія тестера і розробника без централізованої передачі частин проекту допустимо, але тільки у випадку, якщо необхідно терміново перевірити якусь правку. Дуже часто етап розробки і етап тестування взаємозв'язані і йдуть паралельно.

При розробці повинні бути організовані постійно оновлювані сховища готових модулів проекту і бібліотек, які використовуються при збірці модулів. Бажано, щоб процес оновлення сховищ контролювала одна людина. Одне з сховищ повинне бути призначене для модулів, що пройшли функціональне

тестування, а інше — для модулів, що пройшли тестування зв'язків. Перше з них — це чернетки. Друге — те, з чого вже можна збирати дистрибутив системи і демонструвати його замовникові для проведення контрольних випробувань або здачі яких-небудь етапів робіт.

Документація створюється протягом всього процесу розробки. Як тільки модуль пройшов тестування зв'язків, його можна описувати в документації. У випадку якщо модулі змінюються часто, до опису приступають тільки тоді, коли модуль стає більш менш стабільним.

На етапі розробки, як правило, ще раз перевіряється атомарність функцій, а також відсутність їх дублювання.

Завдання створення інформаційної системи в різнорідному середовищі істотно підвищує вимоги до розробників коду і до вибираного засобу розробки. Особливо це стосується розробки системних модулів. Слід приділити увагу модулям, реалізація коду яких залежить від операційної системи. Подібні модулі повинні бути виділені окремо для кожної з операційних систем в групи. Модулі кожної з груп повинні мати строгі інтерфейси обміну — дані, які вони передають і отримують, строго визначені, будь-яке відхилення від специфікації каране. Жоден з модулів поза цією групою не може використовувати ніяких інших викликів, окрім інтерфейсів обміну. Таким чином модулі, залежні від операційної системи, ізолюються від інших модулів [7, с. 380].

Експлуатація перекриває процес тестування, система вводиться в експлуатацію не повністю, а поступово. Введення в експлуатацію проходить три фази:

- первинне завантаження інформації;
- накопичення інформації;
- вихід на проектну потужність.

Первинне завантаження інформації ініціює досить вузький круг помилок - в основному це проблеми розузгодження даних при завантаженні і власні помилки завантажувачів, тобто те, що не було відстежене на тестових даних. Подібні помилки повинні бути виправлені щонайшвидше.

В період накопичення інформації виявиться найбільша кількість помилок, допущених при створенні інформаційної системи. Як правило, це помилки, пов'язані з багатокористувальницьким доступом. Часто на етапі тестування таким помилкам не приділяється належної уваги — мабуть, із-за складності моделювання і дорожнечі засобів автоматизації процесу тестування інформаційної системи в умовах багатокористувальницького доступу. Деякі помилки виправити буде складно, оскільки вони є помилками проектування. Жоден хороший проект від них не застрахований. Це означає, що про всяк випадок треба резервувати час на локалізацію і виправлення таких помилок. Друга категорія виправлень пов'язана з тим, що користувача не влаштовує інтерфейс. Тут не завжди потрібно виконувати абсолютно всі побажання користувача, інакше процес введення в експлуатацію не кінчиться ніколи.

Вихід системи на проектну потужність при вдалому збігу обставин – це виправлення ряду дрібних помилок, і зрідка – помилок серйозних.

Комп'ютерна підтримка методів проектування ІС припускає:

- широке застосування графічних діалогових інтерфейсів (діаграми структур даних, ієрархій функцій, потоків даних і ін.);

- використання комп'ютерних мереж і робота з розподіленими базами даних для підтримки кооперативної групової розробки (використання загальних словників-довідників даних "репозитаріїв");

- поступове розширення використання методів об'єктного моделювання.

Таким чином, використання цього методу, який базується на моделі життєвого циклу «водоспад», забезпечує розробників інформаційних систем зазначеного класу необхідною інформацією про сам процес розробки ІС. Особливістю методу є висока ступінь адаптації до вимог, які сильно і часто змінюються у процесі розробки проекту, тому може успішно використовуватись саме для такого виду проектів, але не може бути рекомендований для проектів, що вимагають дуже високої надійності роботи, тому подальші дослідження будуть спрямовані саме в цьому напрямку.

Отримані результати можуть використовуватись в навчальному процесі для проектування інформаційних систем з використанням CASE-систем.

Література

1. **Вендров А. М.** CASE-технологии. Современные методы и средства проектирования информационных систем. - М: Финансы и статистика, 1998. – 176 с.

2. **Калянов Г.Н.** Case - технологии. Консалтинг при автоматизации бизнес-процессов. - 3-е изд. М.: Горячая линия - Телеком, 2002. - 320 с.

3. **Росс Д.** Структурный анализ (SA): язык для передачи понимания в книге "Требования и спецификации в разработке программ". – М: "МИР", 1994. – С. 46.

4. **Richard Barker,** Case Method: Entity Relationship Modelling, Addison-Wesley, 1998. С. 23.

5. **Макгрегор Джон**, Сайкс Дэвид. Тестирование объектно-ориентированного программного обеспечения: Перевод с английского. – К.: ООО “ТИД ДС”, 2002. – 432 с.

6. **Липаев В.В**. Методы обеспечения качества крупномасштабных программных средств. – М.: СИНТЕГ, 2003. – 520 с.

7. **Липаев В.В**. Обеспечение качества программных средств. Методы и стандарты. Серия “Информационные технологии”. М.: СИНТЕГ, 2001. – 380 с.

Perederiy L.V.

System planning of the informative systems

One of possible methods of development of the informative systems is examined, which is based on the model of life cycle «waterfall», in detail works are investigational on each of the stages of model, the special attention is spared the planning stage, as to most ponderable in this method. A method can be recommended for development of projects of the systems of certain class, during development of which the strong changes of requirements are possible to the project.

Keywords: system planning, model, life cycle, project.

Відомості про автора

Передерій Людмила Василівна – доцент кафедри інформаційних систем і технологій Бердянського університету менеджменту і бізнесу. Основні наукові інтереси зосереджені навколо методів проектування інформаційних систем, CASE-технології, системного аналізу, систем підтримки ухвалення рішень, методів захисту програмного забезпечення.