Проектування програмного забезпечення

Проектування програмного забезпечення

Зміст

Огляд предмету

Цей предмет допоможе студентам отримати практичні навики об’єктно-орієнтованого аналізу, проектування і програмування.
Володіння об’єктно-орієнтованою мовою програмування – це необхідна, але недостатня умова для створення об’єктної системи.

Студенти ознайомляться з процесом проектування за допомогою уніфікованої мови моделювання UML (Unified Modeling Language), шаблонів та швидкого підходу до уніфікованого процесу проектування і програмування. Уніфікований процес є прикладом ітеративної розробки.

UML
Основна увага буде приділятися основам проектування, способам розподілу обов’язків між об’єктами, позначенням мови UML та типічним шаблонам проектування.
UML – це стандарт системи позначень для побудови діаграм. Потрібно не тільки засвоїти UML, але набагато важливіше вивчити принципи ООАП, отримати навички розробки.

Огляд предмету

Прецеденти
Розробка програмної системи в цілому тісно пов’язана з підготовчим етапом – аналізом вимог, який часто включає опис прецедентів (use case).

Визначальні питання при розробці системи:
Як розподілити обов’язки між класами і об’єктами?
Як повинні взаємодіяти об’єкти?
Які функції виконують конкретні класи?

Ітеративна розробка, швидке моделювання та швидкий процес UP
Що повинен робити розробник чи група розробників, щоб забезпечити реалізацію вимог до системи?
Проектування ПЗ необхідно реалізувати в контексті деякого процесу розробки. Пропонується швидкий, легкий та гнучкий ітеративний процес розробки Unified Process (UP).

Огляд предмету

Література:

1. Г. Буч, Д. Рамбо, І. Якобсон Язык UML. Руководство пользователя 2-е издание / Пер. с агнл. – ДМК издательство, 2006 – 496 с.

2. К. Ларман. Применение UML 2.0 и шаблонов проектирования. Практическое руководство 3-е издание / Пер. с агнл. – Издательский дом “Вильямс”, 2013. – 736 с.

Проектування ПЗ

Компанія, що займається виробництвом програмного забезпечення (ПЗ), може отримати успіх тільки у тому випадку, якщо продукція, яка нею випускається має високу якість і розроблена відповідно до вимог користувача. Компанія, котра здатна випускати таку продукцію вчасно і регулярно, при максимально повному і ефективному використанні всіх наявних людських і матеріальних ресурсів буде стабільно існувати.

Отже, основним продуктом такої компанії є першокласне ПЗ, що задовольняє повсякденні вимоги користувача.

Для розробки ефективної програми, котра відповідає своєму призначенню, необхідно **постійно** зустрічатися і працювати з користувачем, щоб з'ясувати реальні вимоги до вашої системи.

Проектування ПЗ

Проектування ПЗ – процес створення ПЗ, а також дисципліна, яка вивчає методи проектування. Проектування ПЗ є окремим випадком проектування продуктів та процесів.

Метою проектування є визначення внутрішніх властивостей системи і деталізації її зовнішніх (видимих) властивостей на основі визначених замовником вимог до ПЗ. Ці вимоги піддаються аналізу.

Спочатку програма розглядається як чорний ящик.

Хід процесу проектування і його результати залежать не тільки від складу вимог, але і вибраної моделі процесу, досвіду проектувальника.

Модель предметної галузі накладає обмеження на бізнес логіку і структури даних.

В залежності від класу ПЗ, що створюється, процес проектування може забезпечуватися як “ручним” проектуванням, так і різноманітними засобами його автоматизації. В процесі проектуванні ПЗ для вираження його характеристик використовуються різні нотації, а також макети.

Проектуванню зазвичай належить:
архітектура ПЗ;
інтерфейси користувача;

Аналіз та проектування

Етап аналізу - дослідження системних вимог і проблеми, а не пошук її вирішення.
Аналіз — включає в себе терміни аналіз вимог (дослідження вимог до системи) та об'єктно-орієнтований аналіз (дослідження об'єктів предметної галузі).

В процесі проектування основна увага приділяється концептуальному рішенню, яке забезпечує виконання основних вимог, але не питанням його реалізації.
Наприклад, на етапі проектування описуються програмні об'єкти чи схема бази даних.
Проектні рішення можуть бути реалізовані в програмному коді.

Як і поняття аналізу, термін проектування також повинно бути уточнено і говорити про об'єктно-орієнтоване проектування чи проектування бази даних.

Об’єкно-орієнтовний аналіз і проектування

В процесі об'єктно-орієнтованого аналізу основна увага приділяється визначенню і опису об'єктів (чи понять) в термінах предметної галузі.

В процесі об'єктно-орієнтованого проектування визначаються програмні об'єкти і способи їх взаємодії з метою виконання системних вимог.

На етапі реалізації чи об'єктно-орієнтованого програмування забезпечується реалізація розроблених компонентів.

Об’єкно-орієнтовний аналіз і проектування

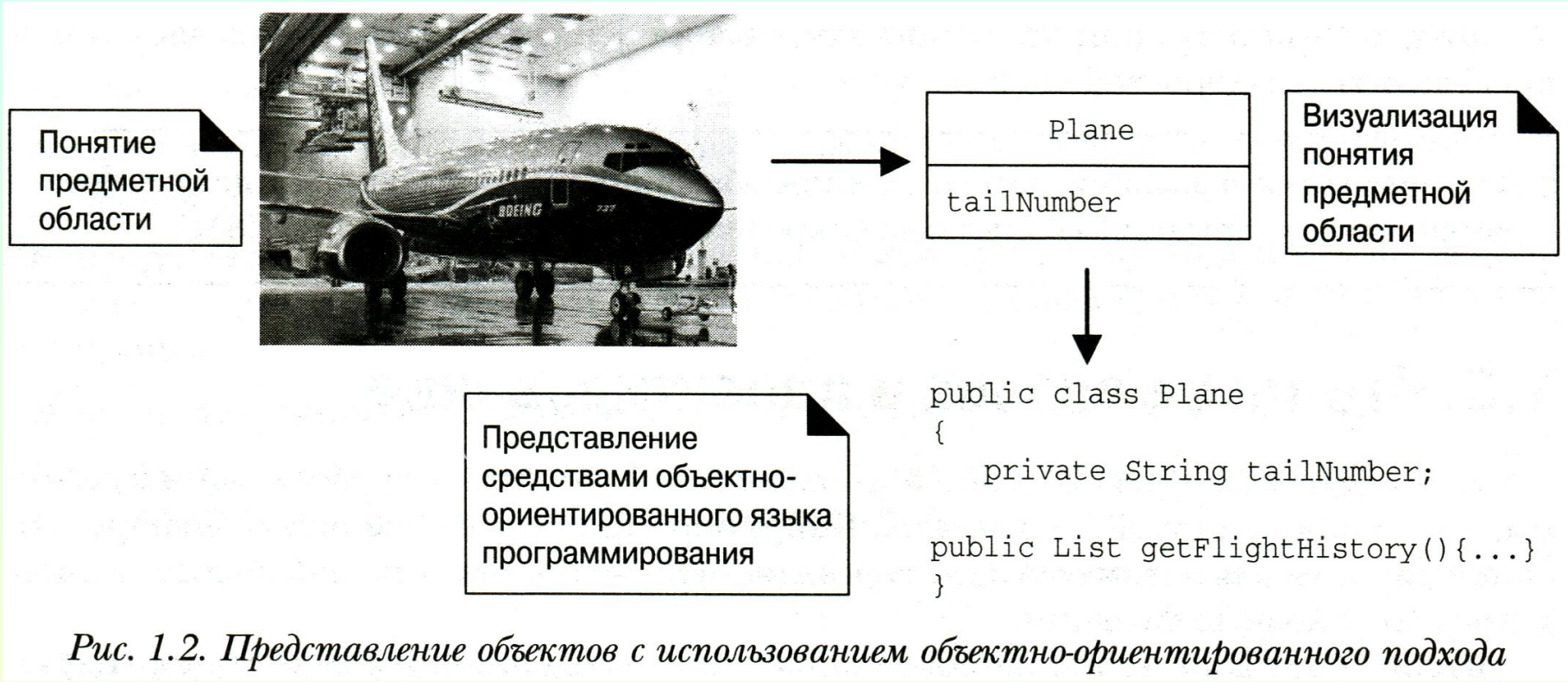


Рис. 1.2. Представление объектов с использованием объектно-ориентированного подхода

Приклад з літаком відпрацювати на практичнрому занятті.

Об’єкно-орієнтовний аналіз і проектування

Послідовність дій процесу розробки програмної системи:

Моделювання та аналіз:
визначити прецеденти;
створити модель предметної галузі;
побудувати діаграми взаємодії;
побудувати діаграми класів проектування.

Архітектура та проектування:
представлення архітектури.

Основні дії при розробці програмної системи

Визначення прецедентів

Аналіз вимог може включати опис процесів чи сценаріїв використання застосування, яке може бути представлено у формі прецедентів. Прецеденти — це просто розповідні історії. Проте вини є популярним засобом аналізу вимог.

Визначення моделі предметної галузі

Об'єктно-орієнтований аналіз пов'язано з описом предметної галузі з точки зору класифікації об'єктів. Модель предметної галузі — це не опис програмних об'єктів, це представлення понятій в термінах реального світу.

Побудова діаграми взаємодії

Створення діаграми класів проектування

Ключові питання об'єктно-орієнтованого проектування

При побудові об'єктних діаграм необхідно відповісти на ключові питання:
Якими є обов'язки об'єктів?
З ким вони взаємодіють?
Які шаблони проектування необхідно застосувати?

Принципи моделювання

Моделювання – це усталена і повсюдно прийнята інженерна методика. Ми будуємо архітектурні моделі будівель, щоб допомогти їх майбутнім мешканцям у всіх деталях уявити готовий об’єкт. Іноді використовується навіть математичне моделювання будівель, щоб врахувати вплив сильного вітру чи землетрусу.

Навряд чи ви зумієте налагодити випуск нових літаків чи автомобілей, не протестувавши свій проект на моделях: від комп’ютерних моделей та креслень до фізичних моделей в аеродинамічній трубі, а згодом і повномасштабних прототипів.
Електричні прибори від мікропроцесорів до телефонних комутаторів також потребують моделювання для кращого розуміння системи і організації спілкування розробників один з одним.

Принципи моделювання

Модель – це упрощене представлення реальності.
Модель – це креслення системи: до неї може входити як детальний план, так і більш абстракте представлення системи “з висоти польоту птиці”.

Гарна модель завжди включає елементи, котрі суттєво впливають на результат, і не вкючає ті, які не мають великого значення на даному рівні абстракції.

Кожна система може бути описана з різних точок зору, для чого використовуються різні моделі, кожна з яких, відповідно, є семантично замкнутою абстракцією системи.

Модель може бути структурованою, що підкреслює організацію системи, або ж поведінковою, відображаючи її динаміку.

Принципи моделювання

Принципи моделювання:
вибір моделі здійснює визначальний вплив на підхід до рішення проблеми і на те, як буде виглядати це рішення;

кожна модель може бути представлена з різним ступенем точності;

кращі моделі – ті, що ближчі до реальності;

неможна обмежуватися створенням тільки однієї моделі. Найкращий підхід при розробці любої нетривіальної системи – використовувати сукупність декількох моделей, які майже не залежать одна від одної.

Визначення моделі предметної галузі

Рис. 1.3. Фрагмент модели предметной области игры в кости

Приклад зі грою в кістки відпрацювати на практичному занятті.

Статичні та динамічні моделі об'єктів

Статичні та динамічні моделі в рамках гнучкої розробки створюються паралельно: спочатку будуються діаграми взаємодії (динамічні), а потім відповідні діаграми класів (статичні).

Саме в процесі динамічного моделювання використовуються шаблони GRASP і реалізується проектування на основі розподілу обов'язків. Ці моменти є ключовими в об'єктно-орієнтованому проектуванні.

Основні дії при розробці програмної системи

Розподіл обов'язків між об'єктами і діаграми взаємодії

Об'єктно-орієнтоване проектування пов'язане з визначенням програмних об'єктів, їх обов'язків і способів взаємодії. Діаграми взаємодії відображають потоки повідомлень між програмними об'єктами та виклики методів.

Розподіл обов'язків між об'єктами і діаграми взаємодії

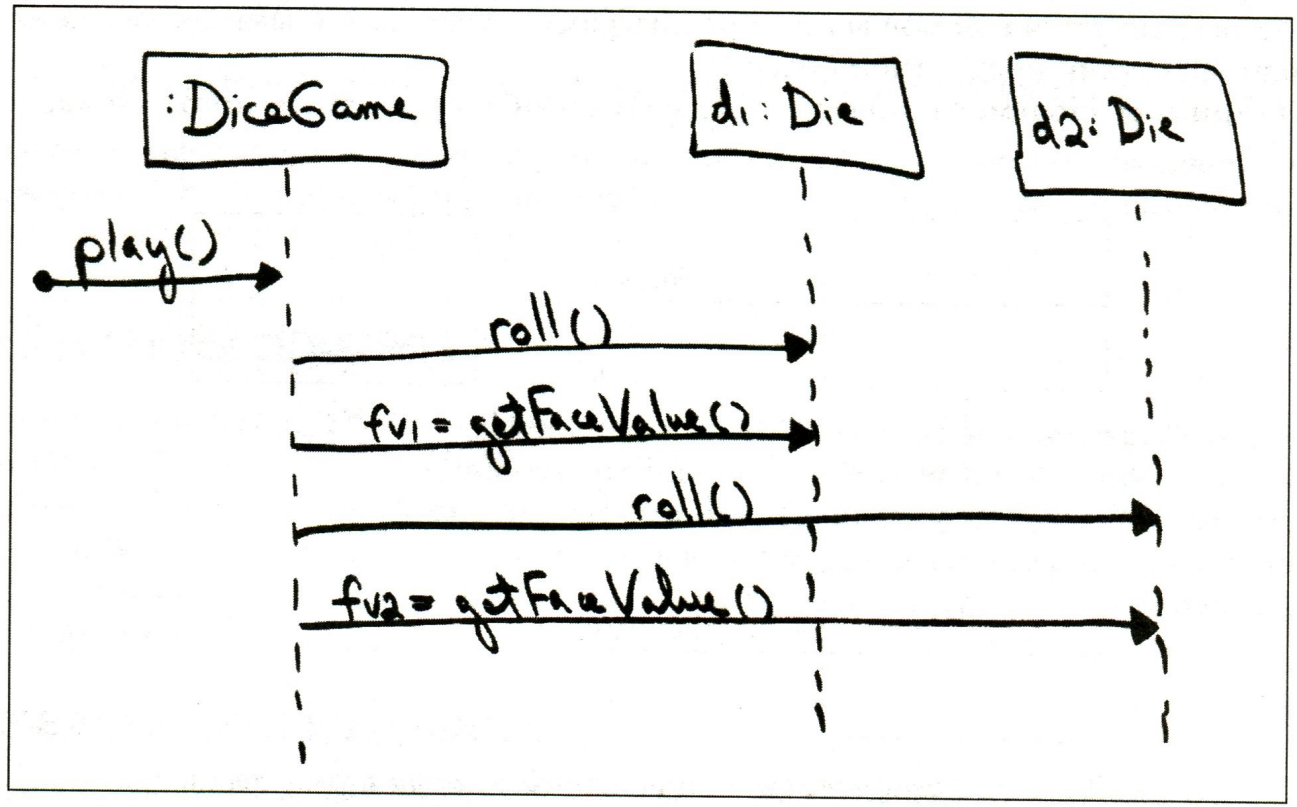


Рис. 1.4. Диаграмма последовательностей, иллюстрирующая передачу сообщений между программными объектами

Пр.: Рис.1.4.

Основні дії при розробці програмної системи

Розробка діаграм класів проектування

Крім динамічного представлення взаємозв'язків об'єктів, які відображаються на діаграмах взаємодії, дуже корисно будувати статичні представлення системи у вигляді діаграми класів проектування.

У відмінності від моделі предметної галузі, які відображають поняття реального світу, ця діаграма описує **програмні** класи.

Основні дії при розробці програмної системи

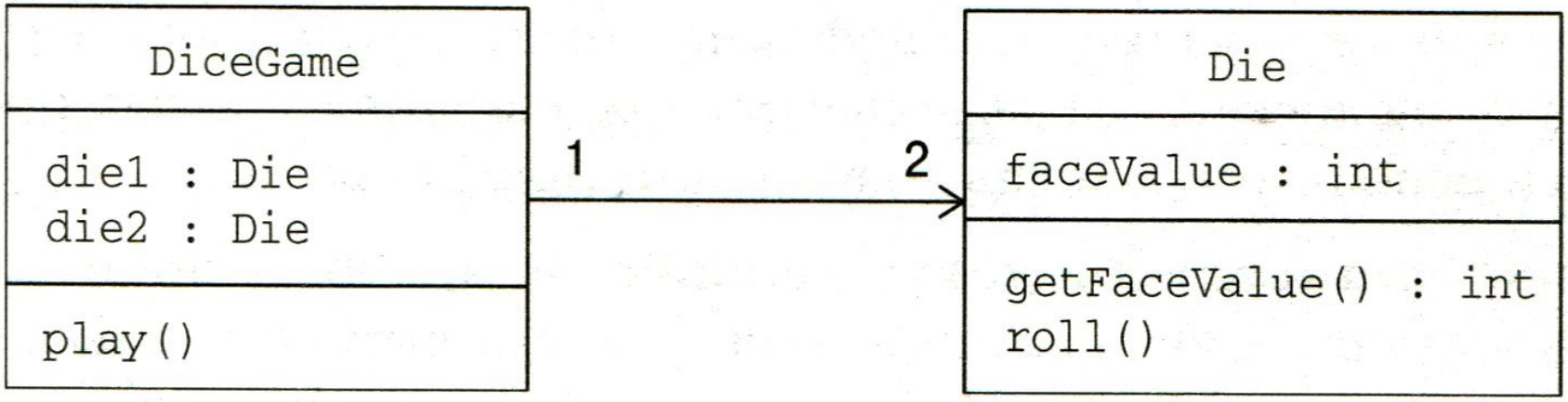


Рис. 1.5. Фрагмент диаграммы классов проектирования

Пр.: Рис.1.5.

Способи об'єктно-орієнтованого проектування

Існує три основні способи об'єктно-орієнтованого проектування:

1. Написання програмного коду.

При цьому відбувається перехід від ментальної моделі до її втілення у програмному коді.

2. Малювання з подальшим коддуванням.

Під цим методом розуміється побудова діаграм UML(Unified Modeling Language) на дошці, чи за допомогою спеціальних CASE-засобів (Computer-Aided Software Engineering), а потім перехід до п.1 з використанням спеціальних інтегрованих середовищ розробки IDE (наприклад: Visual Studio)

3. Малювання.

Способи об'єктно-орієнтованого проектування

Засоби автоматизації розробки програм (CASE-засоби) - інструменти автоматизації процесів проектування та розробки програмного забезпечення для системного аналітика, розробника ПЗ і програміста.

Спочатку під CASE-засобами розумілися тільки інструменти для спрощення найбільш трудомістких процесів аналізу і проектування, але з приходом стандарту ISO / IEC 14102 CASE-засоби стали визначатися як програмні засоби для підтримки процесів життєвого циклу ПЗ.

Що таке UML

Уніфікована мова моделювання UML (Unified Modeling Language) – це візуальна мова для визначення, конструювання та документування артефактів систем.

UML – це стандарт для системи позначень елементів діаграм чи представлень зображень (з елементами тексту), пов’язаних з програмним забезпеченням, переважно об’єктно-орієнтованим.

Способи використання UML

Для чернеток – неповні і неформальні діаграми (найчастіше намальовані від руки на дошці), які створюються для пояснення складних проектних рішень. Використовується міць візуального представлення.

Для створення проектної документації – відносно деталізовані діаграми проектування, які використовуються для візуалізації та кращого розуміння існуючого коду, зворотного проектування (код => діаграми UML) чи генерації коду (діаграми UML => код).

У якості мови програмування – повні специфікації програмних систем на мові UML, які можуть виконуватись. Код, який виконується можна повністю згенерувати.

Швидкий підхід до моделювання (egile modeling) передбачає використання UML для створення чернеток (значна економія часу)

Аспекти використання UML

Концептуальний аспект – діаграми описують сутності реального світу чи предметної галузі.

Аспект (програмної) специфікації – діаграми описують програмні абстракції чи компоненти зі своїми специфікаціями і інтерфейсами, проте без прив'язки до конкретної реалізації (без відповідності C#).

Аспект (програмної) реалізації – діаграми інтерпретуються як опис програмної реалізації на базі конкретної технології чи мови.

Аспекти використання UML

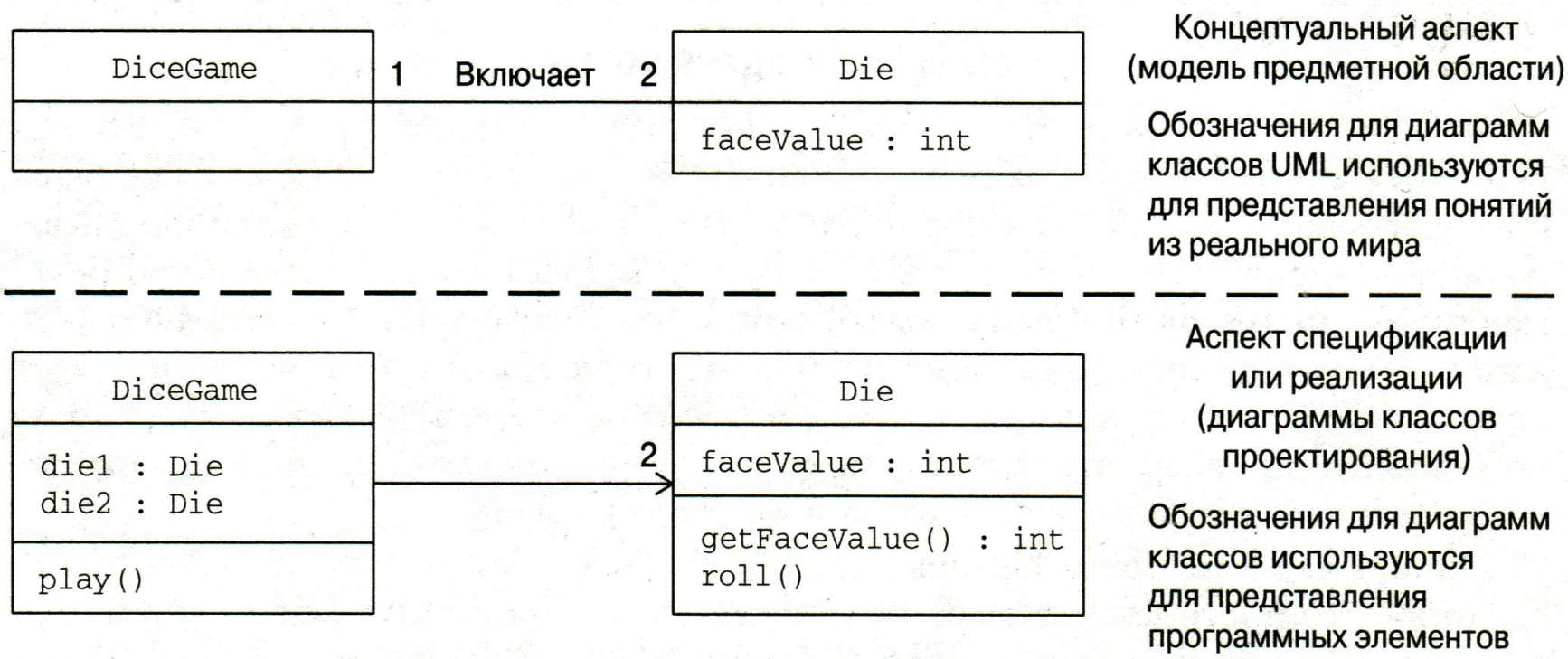


Рис. 1.6. Различные ракурсы использования UML

Пр.: Рис. 1.6.

Значення терміну клас у різних аспектах

Концептуальний клас – поняття із реального світу. В рамках UP модель предметної галузі містить концептуальні класи.

Програмний клас – клас, який представляє специфікацію, чи реалізацію програмного компонента, незалежно від процесу, чи методу.

Клас реалізації – клас, який реалізовано на конкретній ООП мові (C#).

Ітеративний, еволюційний та гнучкий процес

Ітеративна розробка – це основа підходу до створення програмних систем.

Швидке моделювання дозволяє застосувати мову UML найбільш ефективно.

Уніфікований процес – це приклад одного із найпопулярніших ітеративних методів розробки на базі ООАП.

Ітеративний, еволюційний та гнучкий процес

Ітеративна і еволюційна розробка суттєво відрізняється від послідовного, чи каскадного (waterfall), життєвого циклу і передбачає раннє програмування та тестування частин системи в багаторазово повторюваних циклах роботи над проектом.
При використанні такого підходу розробка зазвичай починається ще до детального визначення всіх вимог. При цьому для прояснення та покращення специфікації активно застосовується зворотній зв'язок із зацікавленими особами.

В процесі ітеративної розробки прояснення вимог і проектних рішень в багатьох випадках забезпечується зворотнім зв'язком.
На відміну від цього при використанні каскадного процесу до програмування суттєві зусилля витрачаються на попереднє визначення теоретичних вимог і відпрацювання потенційних проектних рішень.

Опис архітектури

Процес проектування архітектури ПЗ включає в себе збір вимог клієнтів, їх аналіз та створення проекту для компонента ПЗ у відповідність до вимог.

Успішна розробка ПЗ повинна:

забезпечувати баланс неминучих компромісів внаслідок суперечок вимог;

відповідати принципам проектування та рекомендованим методам, вироблених з часом;

доповнювати сучасне обладнання, мережі і системи управління.

Надійна архітектура ПЗ вимагає значного досвіду в теоретичних і практичних питаннях, а також уяви, необхідної для перетворення бізнес-сценаріїв і вимог в надійні і практичні робочі проекти.

Опис архітектури

Архітектура ПЗ включає в себе визначення структурованого рішення, у відповідності до всіх технічних і робочих вимог, одночасно оптимізуючи загальні атрибути якості, такі як продуктивність, безпеку і керованість.
Кожне з цих рішень може значно впливати на якість, підтримуваний і загальний успіх ПЗ.

Сучасне ПЗ рідко буває автономним. Як мінімум, в більшості випадків воно буде взаємодіяти з джерелом даних.
Зазвичай сучасне ПЗ також має взаємодіяти з іншими службами і мережевими функціями для виконання перевірки автентичності, отримання та публікації інформації та надання інтегрованих середовищ для роботи користувачам.

Без відповідної архітектури може бути складно, якщо взагалі можливо, здійснити розгортання, експлуатацію, обслуговування і успішну інтеграцію з іншими системами; крім того, вимоги користувачів не будуть дотримані.

Опис архітектури

Правильне розуміння архітектури забезпечить оптимальний баланс вимог і результатів.
ПЗ з добре продуманою архітектурою буде виконувати зазначені завдання з параметрами вихідних вимог, одночасно забезпечуючи максимально високу продуктивність, безпеку, надійність і багато інших чинників.

На найвищому рівні проект архітектури повинен:
надавати структуру системи, але приховувати деталі реалізації;
охоплювати всі випадки застосування і сценарії;
намагатися враховувати вимоги всіх зацікавлених осіб;
задовольняти настільки, наскільки можливо, всім функціональним вимогам і вимогам до якості.

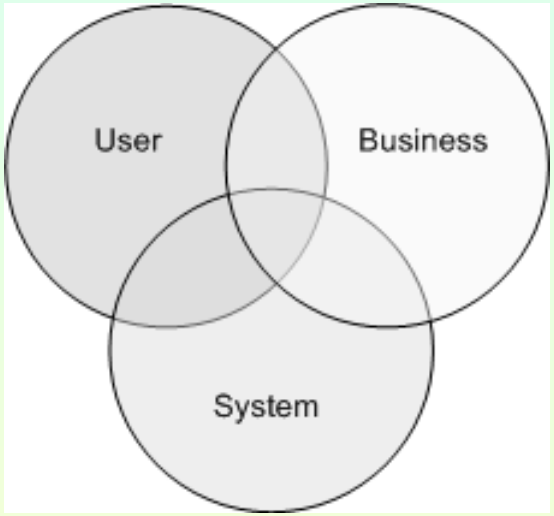
Опис архітектури

Розробник архітектури ПЗ повинен враховувати потреби клієнта.
Однак загальний термін «клієнт» зазвичай складається з трьох суперечних областей відповідальності:
бізнес-вимоги,
вимоги користувача і
системні вимоги.

Бізнес-вимоги зазвичай визначають діапазон таких факторів, як
бізнес-процеси,
фактори продуктивності (такі як безпека, надійність і пропускна здатність), а також
бюджетні обмеження і обмеження на витрати.

Опис архітектури

Вимоги користувача включають в себе
дизайн інтерфейсу,
виробничі можливості і
простоту використання ПЗ.



Системні вимоги:
можливості та обмеження обладнання,
мережі та
середовища виконання.

Розробнику необхідно створити архітектуру, яка підходить для областей, що перекриваються.

Ознайомлення з представленнями архітектури

Архітекторам часто доводиться відповідати на такі питання:
«Як користувачі будуть працювати з додатком?»;
«Як додаток буде розгорнуто у виробничому середовищі і управлятися?»;
«Які вимоги атрибутів якості для програми, такі як безпека, продуктивність, паралелізм, інтернаціоналізація і конфігурація?»;
«Яка повинна бути архітектура програми для забезпечення гнучкості і зручності в обслуговуванні з плином часу?» І
«Які архітектурні тенденції можуть впливати на додаток зараз і після його розгортання?».

Ознайомлення з представленнями архітектури

Остаточний продукт роботи архітектора - це зазвичай набір схем, моделей і документів, що визначають застосування (додаток) з декількох точок зору, щоб при об'єднанні вони могли надати розробникам, групам тестування, адміністраторам і управлінню всю необхідну інформацію для реалізації проекту.

ГОСТ 34. Розробка автоматизованої системи управління (АСУ)

Життєвий цикл процесу створення АСУ згідно ГОСТ 34 (ГОСТ 34.601-90) включає наступні стадії:
Формування вимог до АС
Розробка концепції АС
Технічне завдання
Ескізний проект
Технічний проект
Робоча документація
Введення в дію
Супровід АС

Заключна частина

Якщо ви дійсно хочете створити програмний продукт, за масштабністю замислу порівнянний з житловим будинком чи хмарочосом, то ваша задача не зводиться до написання великого об'єму коду. Насправді проблема в тому, щоб написати правильний код мінімального розміру.

При такому підході розробка якісного ПЗ зводиться до питань вибору архітектури, підходящого інструменту і засобів управління процесом.

Треба мати на увазі те, що багато проектів, які задумані за принципом "собачої будки", швидко розвиваються до розмірів хмарочосу, стаючи жертвою особистого успіху.
Якщо такий ріст не було враховано в архітектурі додатку, технологічному процесі чи при виборі інструментарію, то неминуче настане час, коли виростаючи до розмірів великого будинку будка обрушиться під тягарем своєї ваги.

Проектування програмного забезпечення

Дякую за увагу