УДК 004.415(045)

Кірхар Н.В., к.т.н.

ЗАСТОСУВАННЯ ТЕХНОЛОГІЇ АРХІТЕКТУРНОГО ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Національний авіаційний університет

kirhar@ukr.net

Вступ

Сучасний тренд розвитку напрямку архітектурного проектування, демонструє позитивну динаміку, якщо порівнювати з ситуацією десятирічної давності. Все більше і більше фахівців з галузі інформаційних технологій і бізнес активностей приходять до усвідомлення необхідності спрямованого і обґрунтованого розвитку архітектур програмних продуктів [1].

Програмні продукти які широко використовуються на сьогоднішній день пройшли перевірку часом, змінюючись і допрацьовуючись розробниками. Архітектурне проектування досить нова технологія, при використанні її в той час, можна було б уникнути доробок програмного продукту. У наш час, час високої конкуренції на ринку програмного забезпечення, далі ПЗ, використання архітектурного проектування програмного забезпечення, необхідність [4].

Поточні очікування від технологій, в основі яких інформаційна архітектура, привели до формування нової ідеології в області впровадження і розробки архітектурних продуктів.

В останнє десятиліття дозріло розуміння того, що архітектура програмного забезпечення повинна бути результатом роботи не тільки програмістів, а спеціальних груп різнопрофільних спеціалістів. Ці групи повинні бути здатні з заданим рівнем якості створити повноцінну архітектуру, здатну охопити численні складові інформаційних продуктів.

Постановка проблеми

Для створення інформаційної архітектури необхідно володіти знаннями про вимоги до неї в проектованій області.

Програмні продукти – це основний елемент більшості сучасних високотехнологічних доменів (стільниковий зв'язок, відеотрансляції, охоронна діяльність, керування транспортним та іншими видами руху і т.д.).

Архітектурним проектуванням називають перший етап процесу проектування, на якому визначаються підсистеми, а також структура управління і взаємодії систем.

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 усього часу її експлуатації можна було забезпечити:

– необхідну функціональність системи і ступінь адаптації до мінливих умов її функціонування;

– необхідну пропускну здатність 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. Гарне проектне рішення служить основою високопродуктивної системи.

Основний матеріал

При побудові архітектури мають бути визначені:

- список сценаріїв і прецедентів, критичних з точки зору архітектури;
- можливі проблеми, пов'язані з вибраною архітектурою;
- архітектурні рішення - кандидати, які задовольняють поставленим вимогам і обмеженням.

Для побудови кінцевої архітектури використовується ітеративний підхід, який включає п'ять основних стадій.

1. Визначення цілей архітектури
2. Визначення ключових сценаріїв
3. Складання огляду додатка
4. Ідентифікація ключових проблем
5. Визначення рішення-кандидата
6. Визначення цілей архітектури. Розуміння цілей поточної архітектури дозволяє визначити момент завершення поточної фази і готовності переходу до наступної.
7. Ключові сценарії. Ключові сценарії використовуються для визначення найбільш важливих сторін архітектури і оцінки готовності архітектури - кандидата.
8. Огляд додатка. При створенні огляду визначається тип додатка, архітектурні стилі і технології щоб зв'язати додаток з реальним середовищем де воно функціонуватиме.
9. Ключові проблеми. Ключові проблеми визначають базуючись на атрибутах якості. Ключові проблеми визначають області найбільш вірогідної появи помилки в розробці додатка.
10. Рішення - кандидати. При визначенні рішення кандидата створюються прототипи, які розвивають і покращують поточне прийняте рішення, і, які оцінюються відповідно до ключових сценаріїв, проблем і обмежень.

Процес побудови архітектури використовує підхід ітеративного поліпшення. Перша архітектура - кандидат представляє собою високорівневий дизайн, який повинен перевірятися на відповідність ключовим сценаріям, вимогам, відомим обмеженням, атрибутам якості і відповідності архітектурним рамкам для вибраного архітектурного шаблону.

В процесі поліпшення кандидата з'являються нові деталі, що дозволяє змінювати ключові сценарії, і застосовувати нові підходи до рішення проблем.

Моделі архітектури можуть залежати від функціональних вимог до розроблюваної системі.

• Продуктивність – за критичні операції відповідає якомога менше підсистем - тобто використовується крупномодульна архітектура.

• Захищеність – багаторівнева архітектура системи, найбільш критичні елементи захищені на нижньому рівні.

• Надійність – включаються явно зайві компоненти, які можна змінювати не перериваючи роботу системи.

• Зручність супроводу – архітектура з дрібних компонентів, які можна легко адаптувати під вимоги предметної області.

• Безпека – за всі операції, що впливають на безпеку, має відповідати якомога менше підсистем.

У статичних структурних моделях представлені підсистеми або компоненти, що розробляються в подальшому незалежно [2].

Модель «Репозитарій» заснована на спільному використанні даних. Всі спільно використовувані дані зберігаються в центральній базі даних, доступну всім підсистемам, кожна з яких, в свою чергу, має також власну базу даних. Взаємообмін даними між підсистемами відбувається за допомогою передачі повідомлень.

Переваги: ефективність, централізація засобів управління даними, прозорість моделі спільного використання, багатозадачність.

Недоліки: всі підсистеми повинні бути узгоджені з моделлю сховища даних, проблема розподіленого зберігання сховища, складність переведення вже існуючих систем на цю модель, однакові вимоги безпеки до всіх підсистем.

Модель «Клієнт-сервер» – це модель розподіленої системи, в якій показано розподіл даних і процесів між декількома процесорами. Модель включає три основних компоненти:

1) набір серверів, що надають послуги іншим підсистемам;

2) набір клієнтів, які викликають ці сервіси;

3) мережа, за допомогою якої клієнти отримують доступ до сервісів.

Переваги: простота додавання нових серверів, простота поновлення сервісів.

Недолік, високі вимоги до пропускної здатності мережі.

Модель «Абстрактна машина» організовує систему у вигляді набору рівнів, кожен з яких надає свої сервіси. Кожен рівень визначає абстрактну машину, машинна мову якої (послуги) використовується для реалізації наступного рівня абстрактної машини.

Переваги: покроковий розвиток системи, крос-платформеність.

Недолік, складна структура.

Розробник архітектури повинен організувати підсистеми згідно деякої моделі управління, яка доповнювала б наявну модель структури [1-2]. У моделях управління проектується потік управління між підсистемами. Розглянемо деякі з таких систем.

Централізоване управління. Одна з підсистем повністю відповідає за управління, запускає і завершує роботу інших підсистем. Розрізняють два класи централізованого управління.

1. Модель виклику-повернення – може бути застосована тільки в послідовних системах і реалізує передачу управління "зверху-до низу".

2. Модель диспетчера – застосовується в паралельних системах, в яких системний компонент (диспетчер) координує інші процеси системи, протікаючи паралельно з ними.

Управління, засноване на подіях. Замість однієї підсистеми, відповідальної за управління, на зовнішні події може відповідати будь-яка підсистема. Події, на які реагує система, можуть відбуватися або в інших підсистемах, або в зовнішньому оточенні системи. Тут також виділяють два класи моделей.

1. Передача повідомлень – подія являє собою передачу повідомлення всіх підсистем; будь-яка підсистема, яка обробляє цю подію, відповідає на нього.

2. Переривання – використовуються в системах реального часу.

Після етапу розробки системної структури слідує етап декомпозиції підсистем на модулі. На цьому етапі поширені дві моделі проектування.

Об’єктно-орієнтована модель. У цій моделі система структурована у вигляді сукупності слабо пов'язаних об'єктів з чітко визначеними інтерфейсами. Об'єкти викликають послуги, що надаються іншими об'єктами.

Модель потоків даних. Модулі в даній моделі виконують функціональні перетворення. Дані, що надходять на вхід системи, проходять через всі перетворення і досягають виходу системи.

Поряд з основними моделями, використовуються архітектурні моделі, характерні для конкретної предметної області [3]. Ці моделі називаються проблемно-залежною архітектурою.

Моделі класів систем, відображають класи реальних систем, увібравши в

себе основні характеристики цих класів. Як правило, моделі класів зустрічаються в системах реального часу. Модель компілятора – найбільш відомий приклад цієї моделі.

Базові моделі являють собою ідеалізовану архітектуру, в якій відображені всі особливості, властиві систем, що працюють в даній предметній області. Прикладом такої архітектури може служити модель OSI.

Базові моделі зазвичай не розглядаються в якості методів реалізації; їх основне призначення – служити еталоном для порівняння різних систем в будь-якої предметної області.

Метод проектування ПЗ являє собою організовану сукупність процесів створення ряду моделей, які описують різні аспекти системи, що розробляється з використанням чітко визначеної нотації. На більш формальному рівні метод визначається як сукупність складових:

• концепцій і теоретичних основ, може використовуватися структурний або об'єктно-орієнтований підхід;

• нотації, використовуються для побудови моделей статичної структури і динаміки поведінки проектованої системи, зазвичай використовуються графічні діаграми, і діаграми «сутність - зв'язок» для структурного підходу, діаграми варіантів використання, діаграми класів і ін. - для об'єктно-орієнтованого підходу;

• процедур, що визначають практичне застосування методу (послідовність і правила побудови моделей, критерії, які використовуються для оцінки результатів).

Методи реалізуються через конкретні технології і підтримують їх методики, стандарти та інструментальні засоби, які забезпечують виконання процесів життєвого циклу (ЖЦ) ПЗ.

У даній області вже накопичено певний, найбільш успішний досвід зразків архітектур і архітектурного проектування, які об'єднані в шаблони проектування (design patterns)[1, 2].

У створення архітектурних шаблонів великий внесок вніс класик програмної інженерії Алістер Кокберн [1].

У своїх роботах він представив 15 шаблонів архітектурного проектування. Специфіка полягає в тому, що в них переважно описуються саме чинники «зовнішнього» впливу на архітектури, ніж складові програмної інженерії.

Аналізуючи його роботи, можна виділити 3 стилі застосування шаблонів, які є загальновживаними в процесах проектування.

1. Статичне використання шаблонів. Розробка архітектури відбувається перед початком розробки «коду» майбутнього програмного продукту. Даний шаблон реалізується від початку і до кінця при розробці певної функціональності програми. При застосуванні цього шаблону використовується більше часу і ресурсів на ранніх стадіях з тим, щоб отримати економію при подальшому супроводі та доопрацювання.

2. Еволюційне використання шаблонів. Використання цього шаблону направлено на рівномірний розподіл ресурсів за стадіями проекту розробки програми. В цьому випадку зменшуються «вкладення» на початку проекту, що сприяють отриманню більш швидких початкових результатів у вигляді функціональності, але потрібні додаткові інвестиції при впровадженні шаблону на стадії супроводу / доопрацювання. Такий підхід призводить до того, що пізніше будуть потрібні доопрацювання архітектури та виправлення допущених помилок, виникнення яких неминуче. Цей шаблон має на увазі застосування процесів рефакторінга і реінжинірингу при супроводі та розвитку системи.

3. Невикористання шаблонів. Використання даного шаблону виправдано тільки для невеликого числа проектів. Наприклад, при роботі над старим кодом, в разі вкрай обмежених ресурсів /

термінів в поєднанні з невисокими вимогами до якості / супроводом програмного продукту.

Вибір стилю використання шаблонів відбувається на підставі політики організації, наявних ресурсів і вимог. Вибір робиться системним архітектором на підставі суперечливих даних про майбутнє проекту.

Між теоретичними документами, що описують інформаційні процеси, і їх практичною реалізацією не повинно бути «розриву», як між теорією і практикою.

Документація повинна відображати тільки актуальний стан архітектурних об'єктів і процесів розроблюваного програмного продукту [4].

При описі бізнес-процесів, що становлять рівні архітектури програмного продукту, необхідно врахувати і відобразити такі моменти як.

1. Виконувані бізнес операції.
2. Структура використовуваних даних та її складові.
3. Прототипи екранних форм.
4. Модульність системи і т.д.

На етапі вибору основної практики проектування, застосування якої планується в якості «базису», для створення майбутньої архітектури, необхідно мати уявлення про існуючі методики, які на поточний момент є еталонними:

- модель TOGAF;
- модель Захмана;
- модель Gartner;
- модель META Group.

Використання рекомендацій, наведених в зразкових методиках, допоможе так само вирішити і існуючі організаційні завдання, породжені тим, що в більшості випадків роботами по опису і вдосконаленню внутрішніх бізнес-процесів займаються одні підрозділи (функціональні підрозділи, спеціально створені відділи вдосконалення і якості бізнес-процесів), а реалізують ці вимоги інші (підрозділи інформаційних технологій та ін.).

Частина непорозуміння, в більшості випадків, «знімається» за рахунок використання однієї групи / сімейства нотацій здатних підтримувати перетворення діаграм і моделей, що описують домен бізнес понять і процесів, в кінцевий продукт (код програм). Вибір набору інструментів, а вірніше CASE засобів, в яких реалізована подібна функціональність, повинен забезпечити формування необхідного і достатнього мінімуму моделей і артефактів. Дані атрибути процесу архітектурного проектування повинні підтримувати різні рівні архітектури програмного забезпечення, необхідні для її розробки і подальшого розвитку. Таким чином, можна домогтися того, що вимоги до підтримки і розвитку інформаційної системи будуть доведені до виконання в своєму первісному і істинному розумінні.

Для переходу з етапу опису архітектури бізнес-процесів до формування цілісної IT-архітектури, потрібно додатково формалізувати такі предметні області.

- Архітектура даних.
- Архітектура додатків.
- Архітектура технологій.

Архітектура даних, «зведена» на сутності, виділені після аналізу первинних даних, повинна синтезувати в собі всі елементи інформації, зібраних з опису бізнес процесів. Для опису архітектури даних існує визнана модель опису даних - «сутність-зв'язок» (Entity-Relationship - ER). Діаграма ER чітко структурує всю інформацію, визначаючи структуру таблиць майбутньої бази даних. Подібна властивість призводить до однозначного визначення майбутньої структури даних компанії в прив'язці до існуючих і запланованим бізнес процесів.

Наступна стадія – перехід від архітектури бізнес-процесів і даних до ство-

рення архітектури додатків. Ключовим завданням цієї стадії є визначення залежності між вищерівневою архітектурою бізнес-процесів і додатками, які за допомогою своїх компонентів і модулів повинні забезпечити необхідний рівень автоматизації процесів підприємства. На моделі, що описує даний тип архітектури, повинні бути розташовані основні інформаційні системи, з подальшою декомпозицією до рівня прототипів екранних форм, за допомогою яких здійснюється взаємодія з користувачами систем.

У тих випадках, коли для формування архітектури необхідне впровадження і використання безлічі різнорідних додатків, що підтримують конкретні бізнес процеси, доречно говорити про застосування «карти підтримки процесів інформаційними системами», документ який дозволяє вищерівнево представляти і відслідковувати весь архітектурний ландшафт організації.

Після етапу створення архітектури додатків настає етап реалізації архітектури технологій, які являють собою елементи майбутньої IT-інфраструктури.

• Сервера, на яких будуть розміщені бази даних додатків і модулі архітектури, що підтримують логіку обчислень.

• Мережеві елементи, маршрутизатори потоків вхідного / вихідного трафіку, ін. функції.

• Обладнання, необхідне для підтримки функціонування додатків.

Висновок

Виконано порівняльний аналіз моделей. Багато робіт фахівців в області створення надійних і високопродуктивних архітектур, спрямовані на те, щоб продемонструвати високу ступінь впливу архітектур програмних продуктів і комерційну успішність переважної більшості сучасних організацій.

При досягненні прозорості та взаємозв'язку архітектури бізнес-процесів, даних, додатків і технологій можна говорити про створення бази для побудови загально корпоративної системи управління змінами та типізації вимог до змін інформаційних систем.

Наведено повний опис застосування технології архітектурного проектування при розробці комплексів інформаційних технологій, що позитивно позначиться на розвитку в цілому домену інформаційних технологій.

Список літератури

1. Алистер Кокберн. Современные методы описания функциональных требований к системам. / Алистер Кокберн – М.: Лори, 2016. – 264 с.

2. Руководство Microsoft по проектированию архитектуры приложений. Онлайн-книга: Изд-во Microsoft, 2009. – 529 с.

3. Zachman A. A. Framework for Information Systems Architecture / Zachman A. A. // IBM Systems Journal. – 1987. – Vol. 26. No 3.

4. Кірхар Н.В. Аналіз технологій проектування інформаційних систем. / Кірхар Н.В. // Проблеми інформатизації та управління: зб. наук. праць – Київ – 2015. – №4 (52). – С.45-49.

Кірхар Н.В., к.т.н.

ЗАСТОСУВАННЯ ТЕХНОЛОГІЇ АРХІТЕКТУРНОГО ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Багато робіт фахівців в області створення надійних і високопродуктивних архітектур, спрямовані на те, щоб продемонструвати високу ступінь впливу архітектур програмних продуктів і комерційну успішність переважної більшості сучасних організацій. Розглянуто концепцію та методологію застосування архітектурного підходу при створенні комплексних інформаційних систем, визначені переваги та недоліки. Розглядаються питання вибору та реалізації різних архітектур інформаційних систем, склад та взаємозв'язок процесів при розробці. Розробник архітектури повинен організувати підсистеми згідно деякої моделі управління, яка доповнювала б наявну модель структури. У моделях управління проектується потік управління між підсистемами. Гарне проектне рішення служить основою високопродуктивної системи

Ключові слова: архітектурне проектування, інформаційні системи, проект

Kirhar N.V., Ph.D.

SOFTWARE ARCHITECTURAL DESIGN TECHNOLOGY APPLICATION

Many specialists work in the field of creating reliable and high-performance architectures aimed at demonstrating the high degree of influence of software architectures and the commercial success of the overwhelming majority of modern organizations. The concept and methodology of application of architectural approach at creation of complex information systems, advantages and disadvantages are considered. The questions of choice and implementation of different architectures of information systems, composition and interrelation of processes in development are considered. The developer of architecture must arrange the subsystem according to a certain management model that would complement the existing structure model. In the control models, the flow of control between the subsystems is projected. A good design solution serves as the basis for a highly productive system

Keywords: architectural design, information systems, project.