ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

УДК 004.891

д.т.н., доц. **Гунченко Ю.О.** (ОНУ)
к.т.н., доц. **Джулій В.М.** (ХмНУ)
к.т.н., доц. **Красильников С.Р.** (ХмНУ)
Солодєєва Л.В. (ВІКНУ)
Чешун Д.В. (ХмНУ)

МЕТОДИ АНАЛІЗУ ТА СИНТЕЗУ РОЗРОБКИ WEB - ДОДАТКІВ

У статті запропонований «ресурсний» метод організації контролерів веб-додатка.

Архітектура програмного забезпечення (комп'ютерної системи) являє собою сукупність структур, що складаються з програмних елементів, зовні видимих властивостей цих елементів та взаємозв'язків між ними. Архітектура – це незмінна глибинна структура веб-додатка. Помилки, закладені в архітектуру при її проектуванні, будуть призводити до ще більших помилок в процесі її реалізації (кодування).

Найперспективнішим на даний момент методом розробки веб-додатка є метод, заснований на парадигмі «Модель-Представлення-Контролер». Однак цей метод не позбавлений недоліків, найістотнішим з яких є відсутність методів структурного синтезу програмного коду рівня моделей, рівня представлення і рівня контролерів. У зв'язку з цим поширеним явищем стало невірне трактування архітектури, що призвело до появи великої кількості не уніфікованого програмного коду, що не задовольняють вимогам якості.

Для усунення розглянутих недоліків пропонується модель архітектури веб – додатка і метод структурного синтезу архітектури, а також вихідного коду рівня моделей, рівня представлення і рівня контролерів. Синтез здійснюється методом угрупування сутностей предметної області. Сутності веб-додатка необхідно представити спеціальними класами - моделями. Сукупність моделей становить шар доступу до даних. У запропонованому методі модельний шар охоплює меншу множину функцій, що реалізовуються, ніж модельний шар в існуючій архітектурі «Модель-Представлення-Контролер» за рахунок введення сервісного шару.

На основі визначень рівнів абстракції стає можливим сформувати визначення модуля веб-додатка. Модуль веб-додатка - сукупність представлень (шаблонів і видів), контролерів, моделей даних і базових моделей, сервісів, конфігурацій і тестів, віднесених до певної частини предметної області додатка.

Запропонований «ресурсний» метод організації контролерів веб-додатка. Суть методу полягає в тому, що необхідно проводити розробку рівня контролерів, базуючись не на моделях і представленнях, а на сутностях, над якими планується проводити операції. Такі сутності будуть виступати ресурсами веб-додатка.

Використання пропонованої архітектури і методик розробки веб-додатка дозволить покращити показники якості і разом з тим збільшити продуктивність розробки.

Ключові слова: веб-додатки, контролери, моделі, представлення, програмний код, метод, алгоритм, структурний синтез, інтернет-технології.

Вступ. В процесі розробки програмного забезпечення постають проблеми якості, вартості та надійності роботи системи. Деяке програмне забезпечення містить мільйони рядків коду, які, як і очікується, будуть правильно виконуватись в змінних умовах. Протягом декількох десятиліть вирішується задача пошуку повторюваного, передбаченого процесу або методології, яка б покращала продуктивність, якість і надійність розробки програмного забезпечення. Для коректної постановки задачі, перш за все, необхідно сформулювати опис предметної області, її характеристик і одиниць вимірювання. В процесі створення або модифікації веб-додатка розробник виконує роботу, яка характеризується трудомісткістю, що виражається (вимірюється) в *sp (story point)*. Для оптимізації процесу розробки програмного забезпечення необхідною умовою є оптимізація процесу роботи всіх програмістів – членів команди.

Постановка задачі. Архітектура програмного забезпечення або комп'ютерної системи являє собою сукупність структур, що складаються з програмних елементів, зовні видимих властивостей цих елементів та взаємозв'язків між ними. «Зовні видимими властивостями» називаються очікування інших елементів щодо даного елемента: наданий сервіс, характеристики швидкодії, механізм обробки помилок, використання загальних ресурсів і т.д. Архітектура – це незмінна глибинна структура веб-додатка, помилки, закладені в архітектуру при її проектуванні, будуть призводити до ще більших помилок в процесі її реалізації (кодування).

Найперспективнішим на даний момент розробки веб-додатка є метод розробки, заснований на парадигмі «Модель-Представлення-Контролер». Це підтверджує кількість веб-додатків, побудованих на його основі. Однак цей метод не позбавлений недоліків, найістотнішим з яких є відсутність методів структурного синтезу програмного коду рівня моделей (М-рівня), рівня представлення (П-рівня) і рівня контролерів (К-рівня). У зв'язку з цим поширеним явищем стало невірне трактування архітектури, що призвело до появи великої кількості не уніфікованого програмного коду, що не задовольняють вимогам якості.

Основна частина. Для усунення розглянутих недоліків пропонується модель архітектури веб – додатка (рис. 1) і метод структурного синтезу архітектури веб-додатка, а також вихідного коду рівня моделей, рівня представлення і рівня контролерів. Модель заснована на архітектурі веб – додатка з виділеною конфігурацією і двоетапним завантаженням.

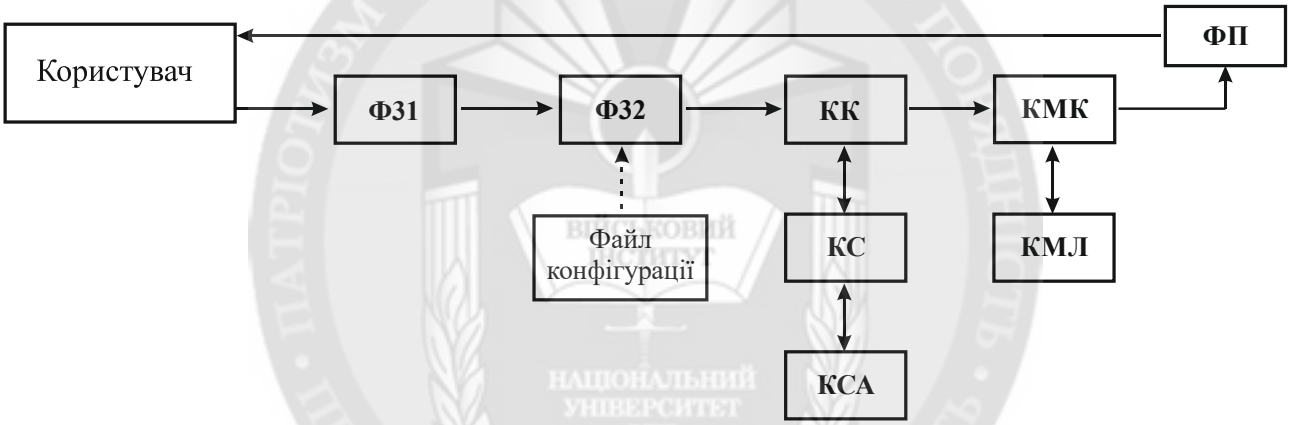

Рис. 1. Модель архітектури веб – додатка

КК - рівень пропонується піддати декомпозиції на три рівні: власне КК - рівень (в якому залишаються «тонкі» контролери), КС- рівень (в якому знаходяться сервіси) і КСА - рівень (в якому розташовуються сервіс-адаптери). КМ-рівень пропонується розділити на рівень концептуальної (КМК) і логічної моделей (КМЛ).

В процесі розробки інформаційної системи обсяг програмного коду має тенденцію до збільшення, що дозволяє здійснити його класифікацію за певною ознакою. Пропонується розділити множину функцій веб-додатка на декілька підмножин, що не перетинаються за функціональною ознакою (за призначенням). Розподіл буде здійснюється методом угрупування сутностей предметної області. Сутності веб-додатка необхідно представити спеціальними класами - моделями. Сукупність моделей становить шар доступу до даних. У запропонованому методі модельний шар охоплює меншу множину функцій, що реалізовуються, ніж модельний шар в існуючій архітектурі «Модель-Представлення-Контролер» за рахунок введення сервісного шару. За характером використання рівні (рис. 2) діляться на виконувані (безпосередньо беруть участь у формуванні вихідної дії веб-додатка), і не виконувані (використовуються для службових цілей). Розгляд виконуваних рівнів починається від самого низькорівневого шару. Рівень зберігання даних складається з класів, іменованих як «базові моделі даних», де однозначно відображаються сутності, з якими працює програма. Базові моделі безпосередньо не використовуються в додатку.

Рівень доступу до даних складається з класів, які наслідуються від базових. При цьому

вони можуть містити функції роботи з екземплярами моделі. Рівень логіки предметної області складається із сукупності класів, які називаються контролерами, функції яких називаються діями.

Рівень сервісів складається з функцій взаємодії з групою моделей, пошуку, модифікації і взаємодії з ними. Такі функції називаються сервісними. Для реалізації шаблону проектування «стратегія» в рамках сервісів застосовуються класи - сервіс-адаптери. Шаблон «стратегія» - поведінковий шаблон проектування, призначений для визначення сімейства алгоритмів, інкапсуляції кожного з них і забезпечення їх взаємозамінності. Його використання дозволяє вибирати алгоритм шляхом визначення відповідного класу.

Рис. 2. Рівні абстракції веб-додатка

Шаблон «Стратегія» дозволяє змінювати обраний алгоритм незалежно від об'єктів-клієнтів, які його використовують. Крім цього шару, адаптери можуть використовуватися в шарі зберігання даних для інкапсуляції особливостей доступу до різних баз даних. Такі адаптери називаються адаптерами баз даних. Будь-який програмний код, який безпосередньо взаємодіє з сервером бази даних (відкриття з'єднань, робота з транзакціями і т.д.), повинен бути розміщений в адаптері відповідного програмного сервера бази даних.

Рівень представлення складається з файлів, іменованих шаблонами (каркасами) і представленнями. Шаблони є представленнями верхнього рівня, в них може бути включений результат обробки («рендеринга») представленнь, який полягає в заповненні динамічних представлень їх значеннями. Динамічні представлення мають локальну область видимості і доступні тільки всередині представлення. Цей підхід реалізує принцип інкапсуляції даних. Представлення можуть включати в себе результат рендеринга інших представлень з необмеженою вкладеністю. Однак на практиці рекомендується застосовувати не більше 3-х рівнів рендеринга представлень в зв'язку з витратами процесорного часу на їх рендеринг. Багаторазово використовувані представлення з невеликими ділянками HTML-розмітки називаються частковими представленнями, а представлення, оброблювані в циклі по масиву змінних - циклічними.

Рівень конфігурації і супроводу реалізує завдання супроводу програмного коду, налаштування параметрів роботи веб-додатка і включає в себе конфігураційні файли. При великій кількості модулів в інформаційній системі рекомендується використовувати окремі конфігурації для кожного модуля, в іншому випадку, оптимальним рішенням буде використання одного файлу з його секціонуванням або додаванням префіксів для налаштувань певного модуля. Налаштування в файлі обов'язково повинні бути згруповані. Окрім групування по модулям, можливе використання групування по підсистемах, однак це може привести до великого обсягу налаштувань в секції файлу.

Рівень тестування та аудиту призначений для верифікації веб-додатка. Це невиконуваний рівень, так як ні при завантаженні, ні при виконанні додатка файли цього рівня не використовуються. Однак він необхідний для проведення самотестування програмного коду. Він зберігає в собі тестові контролери, а також набір визначених значень сутностей (моделей) – «фікстур», які присутні в пам'яті тільки під час тестування коду. Також до цього рівня відносяться журнали веб-додатка, такі як: журнал виконання і журнал тестування. На основі визначень рівнів абстракції стає можливим сформувати визначення модуля веб-додатка. Модуль веб-додатка - сукупність представлень (шаблонів і видів), контролерів, моделей даних і базових моделей, сервісів, конфігурацій і тестів, віднесених до певної частини предметної області додатка. Для вирішення завдання функціонального тестування запропонована архітектура може бути модифікована наступним чином (рис. 3).

Залежно від джерела запиту (К - користувач або Т – система тестування) основним завантажником (ФЗ2) приймається рішення, направляти запит в загальне МПК дерево класів, або ж підключити попередньо тестуючий контролер (КТ). Він є «обгорткою» над контролером веб-додатка, що тестується, (одним із множини КК - класів контролерів) і може підключати ті ж моделі, що і контролери з множини КК.

При проектуванні веб-додатка виникає питання про доцільність розмежування програмного коду за рівнями абстракції на початкових етапах розробки. Життєвий цикл веб-додатка складається з декількох ключових етапів: проектування, прототипування, розробка веб-додатка, тестування і налагодження, розгортання, експлуатація та зняття з експлуатації.

На етапі проектування веб-додатка розробник повинен визначити початковий розподіл. Він характеризується тим, що при його виконанні може бути не повністю визначений набір сутностей предметної області, що відображаються на класах моделей. Ця невизначеність переноситься також на класи сервісів (здійснюють маніпуляції з моделями), контролерів (здійснюють виклики функцій сервісів), представлень і т.д. Перенесення визначеності характеризується властивостями взаємозалежностей різних частин веб-додатка.

Процес формування життєвого циклу веб-додатка пов'язаний з процесами, що відбуваються на М-, К- і П-рівнях веб-додатка. На даний момент існує декілька методів розробки класів К - рівня, проте вони володіють тими чи іншими недоліками. Основний недолік пов'язаний з тим, що розробнику необхідно вибрати частину прототипу веб-додатка, на якому необхідно зосередитись при проектуванні і кодуванні К-рівня: П -рівень (відображення призначеного для користувача інтерфейсу на контролери) або ж М-рівень (відображення концептуальних моделей предметної області на контролери). Будь-який вибір

- це компроміс з його недоліками; в разі відсутності вибору буде відсутня і чітка архітектура, що неминуче призведе до розробки програмного коду, що не відповідає вимогам якості.

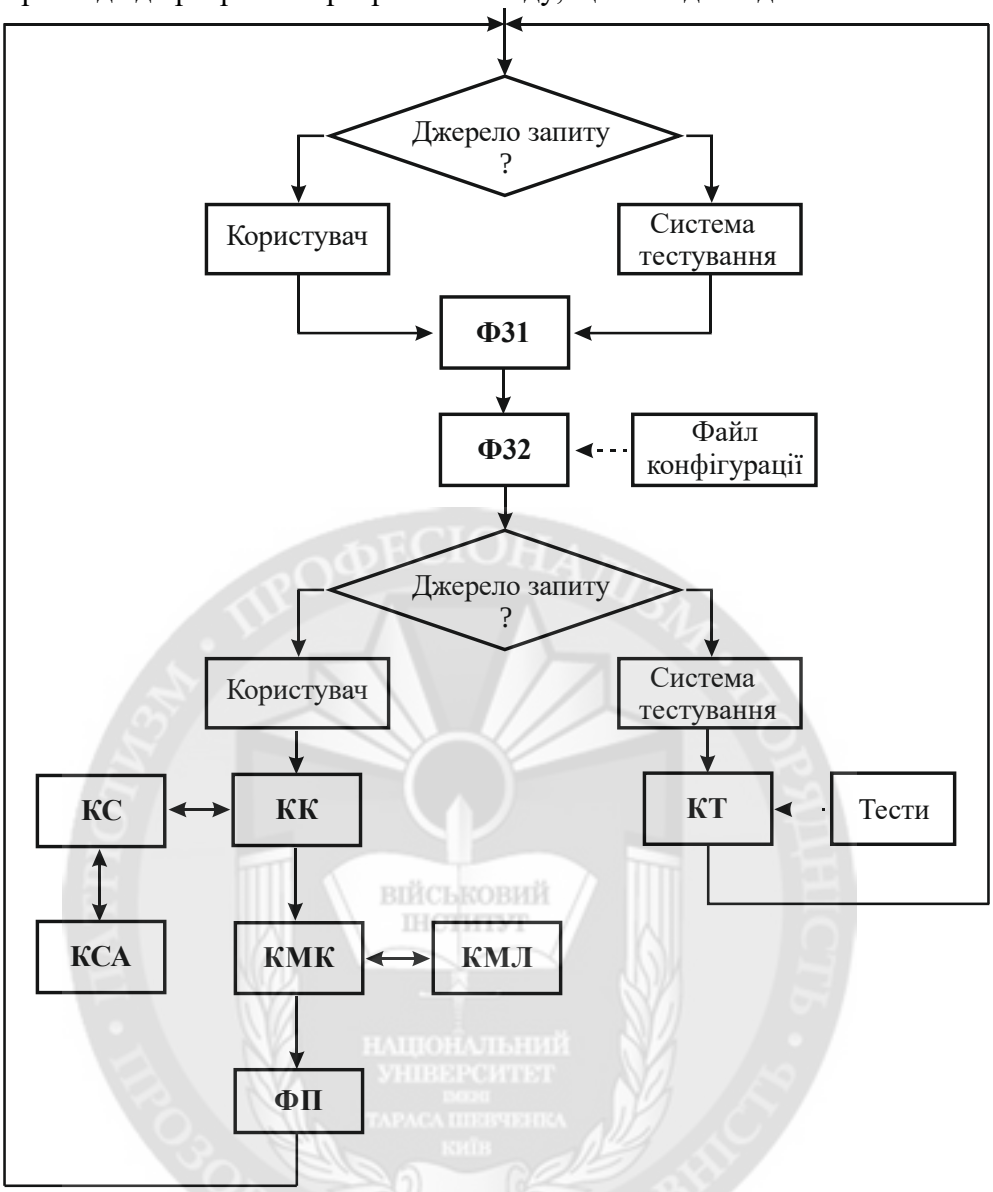

Рис. 3. Архітектура веб-додатка з елементами верифікації

Існують два основних методи розробки зовнішнього програмного інтерфейса веб-додатка: віддалений виклик процедур (RPC) і передача стану представлення (REST). Метод RPC характеризується більшою кількістю методів, ніж ресурсів. А метод REST - навпаки, великою кількістю ресурсів і меншою кількістю методів. Мережний ресурс - концептуальна модель предметної області, доступна по мережі разом з функціями доступу до неї. В ідеальному веб-додатку, побудованому за REST підходом, при необхідності додавання нового ресурсу (контролера) додається фіксована кількість методів цього ресурсу. При реальній розробці за REST підходом спочатку проходить етап відображення моделей на контролери, а після цього поповнення методів. У разі розробки по RPC-підходу спочатку додається множина методів і ресурсів, а потім процес додавання стабілізується. Характер зростання кількості ресурсів і функцій на початковому етапі важливий тому, що в цей період закладається основа архітектури веб-додатка, і вартість помилки (необхідний час на її усунення) в цей період вища, ніж в подальшому.

Запропонований «ресурсний» метод організації контролерів веб-додатка (рис. 4). Суть методу полягає в тому, що необхідно проводити розробку К-рівня, базуючись не на моделях, і не на представленнях, а на сутностях, над якими планується проводити операції. Такі

сутності будуть виступати ресурсами веб-додатка, а зовнішній програмний інтерфейс до ресурсу буде називатися контролером ресурсу. Ресурс може бути моделлю, представленням або ж сутністю, яка потребує зберігання в БД (наприклад, сесії користувача). Також ресурсом може виступати група сутностей, або ж група ресурсів. У такому випадку користувач буде взаємодіяти з мета-ресурсом.

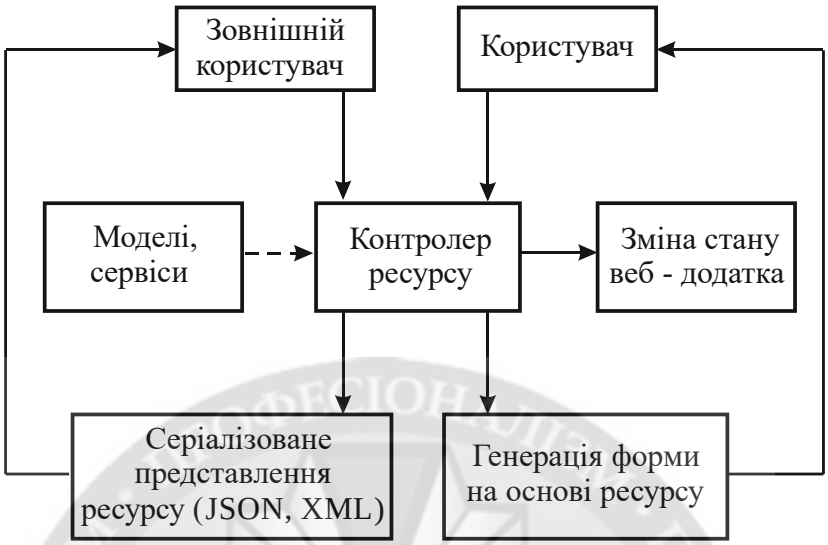

Рис. 4. Контролер ресурсу веб-додатка

Крім того, при такому підході К-рівень може контролювати представлення ресурсу в залежності від того, від кого отримано запит. Для користувача повертати HTML-представлення, а для зовнішнього користувача (ЗК) ($ЗК_2$ рис. 4) його серіалізований варіант в одному із стандартних форматів (JSON, XML).

Висновки. Архітектура програмного забезпечення (комп'ютерної системи) являє собою сукупність структур, що складаються з програмних елементів, зовні видимих властивостей цих елементів та взаємозв'язків між ними. Найперспективнішим на даний момент розробки веб-додатка є метод розробки, заснований на парадигмі «Модель-Представлення-Контролер». Однак цей метод не позбавлений недоліків, найістотнішим з яких є відсутність методів структурного синтезу програмного коду рівня моделей, рівня представлення і рівня контролерів. У зв'язку з цим поширеним явищем стало невірне трактування архітектури, що призвело до появи великої кількості не уніфікованого програмного коду, що не задовольняють вимогам якості. Для усунення розглянутих недоліків пропонується модель архітектури веб – додатка і метод структурного синтезу архітектури веб-додатка, а також вихідного коду. Синтез здійснюється методом угрупування сутностей предметної області. У запропонованому методі модельний шар охоплює меншу множину функцій, що реалізовуються, ніж модельний шар в існуючій архітектурі «Модель-Представлення-Контролер» за рахунок введення сервісного шару.

Запропонований «ресурсний» метод організації контролерів веб-додатка. Суть методу полягає в тому, що необхідно проводити розробку рівня контролерів, базуючись не на моделях і представленнях, а на сутностях, над якими планується проводити операції.

Використання пропонованої архітектури і методик розробки веб-додатка дозволить покращити показники якості і разом з тим збільшити продуктивність розробки.

ЛІТЕРАТУРА:

1. Майк Кон. Scrum: Гибкая разработка ПО. / Майк Кон. – Изд-во: Диалектика-Вильямс, 2016. – 576 с.

2. Лєнков С.В. Концептуальна схема системи інтелектуальної обробки даних / С.В. Лєнков, В.М. Джулій, О.М. Горбатюк, Н.М. Берназ // Збірник наукових праць Військового інституту

Київського національного університету імені Тараса Шевченка. – К.: ВІКНУ, 2014. – Вип. № 46. – С.181-190

3. Джулій В.М. Методи та алгоритми розробки web-додатків / В.М. Джулій, Ю.О. Гунченко, Д.В. Чешун // Збірник наукових праць Військового інституту Київського національного університету імені Тараса Шевченка. – К.: ВІКНУ, 2017. – Вип. № 56. – С.107-115

4. Роберт Мартин Быстрая разработка программ. Принципы, примеры, практика. /Роберт Мартин, Джеймс Ньюкирк, Роберт Косс – Изд-во: Диалектика-Вильямс, 2004. – 752 с.

5. Сэм Руби. Rails 4. Гибкая разработка веб-приложений / Сэм Руби, Дэйв Томас, Дэвид Ханссон – Изд-во: Питер, 2014. – 448 с.

6. Роберт Мартин Гибкая разработка программ на Java и C++. Принципы, паттерны и методики. /Роберт С. Мартин, Джеймс Ньюкирк, Роберт Косс – Изд-во: Диалектика-Вильямс, 2016. – 704 с.

7. Гарднер Л. Разработка веб-сайтов для мобильных устройств; / Л. Гарднер, Д.Григсби – Питер - Москва, 2013. - 448 с.

8. Дакетт Джон HTML и CSS. Разработка и дизайн веб-сайтов (+ CD-ROM); / Джон Дакетт – Эксмо - Москва, 2013. - 480 с.

9. Китинг Flash MX. Искусство создания web-сайтов; / Китинг, Джоди –ТИД ДС - Москва, 2012. - 848 с.

10. Кузнецов М. PHP. Практика создания Web-сайтов; / М. Кузнецов, И.Симдянов–БХВ-Петербург - Москва, 2012. - 347 с.

11. Митчелл 5 проектов Web-сайтов от фотоальбома до магазина; / Митчелл, Скотт –М.: НТ Пресс - Москва, 2013. - 224 с.

12. Фрейен Бен HTML5 и CSS3.Разработка сайтов для любых браузеров и устройств; / Бен Фрейен– Питер - Москва, 2014. - 304 с.

13. Чебыкин Ростислав Разработка и оформление текстового содержания сайтов; / Ростислав Чебыкин –БХВ-Петербург - Москва, 2014. - 528 с.

14. Энж Эрик. SEO. Искусство раскрутки сайтов; / Эрик Энж , Стефан Спенсер , Рэнд Фишкин , Джесси Стрикчиола –БХВ-Петербург - Москва, 2014. - 668 с.

REFERENCES:

1. Majk Kon. Scrum: Gybkaja razrabotka PO. / Majk Kon. – Izdatelstvovo: Dyalektyka-Vyl'jams, 2016. – P. 576.

2. Ljenkov S.V. Konceptual'na shema systemy intelektual'noi' obrobky danyh / S.V. Ljenkov, V.M. Dzhulij, O.M. Gorbatjuk, N.M. Bernaz // Zbirnyk naukovyh prac' Vijs'kovogo instytutu Kyi'vs'kogo nacional'nogo universytetu imeni Tarasa Shevchenka. – K.: VIKNU, 2014. – Pub. № 46. – P.181-190

3. Dzhulij V.M. Metody ta algorytmy rozrobky web-dodatkiv / V.M. Dzhulij, Ju.O. Gunchenko, D.V. Cheshun // Zbirnyk naukovyh prac' Vijs'kovogo instytutu Kyi'vs'kogo nacional'nogo universytetu imeni Tarasa Shevchenka. – K.: VIKNU, 2017. – Pub. № 56. – P.107-115

4. Robert Martyn Bistraja razrabotka programm. Pryncypi, prymeri, praktika. /Robert Martyn, Dzhejms N'jukyrk, Robert Koss – Izdatelstvo: Dyalektyka-Vyl'jams, 2004. – 752 p.

5. Sem Ruby. Rails 4. Gybkaja razrabotka veb-prylozhenyj / Sem Ruby, Dejv Tomas, Devyd Hansson – Izdatelstvo: Pyter, 2014. – 448 p.

6. Robert Martyn Gybkaja razrabotka programm na Java y C++. Pryncypi, patterni, metodyki. /Robert S. Martyn, Dzhejms N'jukyrk, Robert Koss – Izdatelstvo: Dyalektyka-Vyl'jams, 2016. – 704 p.

7. Gardner L. Development of websites for mobile devices; / L. Gardner, D. Grigsbi – Peter - Moscow, 2013. - 448 p.

8. Dakett John of HTML and CSS. Development and design of websites (+ CD ROM); / John Dakett – Eksmo - Moscow, 2013. - 480 p.

9. Keating Flash MX. Art of creation of the websites; / Keating, Jodi – TID DS - Moscow, 2012. - 848 p.

10. Kuznetsov M. of PHP. Practice of creation of the Websites; / M. Kuznetsov, I. Simdyanov – BHV-St. Petersburg - Moscow, 2012. - 347 p.

11. Mitchell of 5 projects of the Websites from a photo album to shop; / Mitchell, Scott – M.: NT the Press - Moscow, 2013. - 224 p.

12. Freyen Ben of HTML5 and CSS3. Razrabotka of the websites for any browsers and devices; / Ben Freyen – Peter - Moscow, 2014. - 304 p.

13. Chebykin Rostislav Razrabotka and execution of the text contents of the websites; / Rostislav Chebykin – BHV-St. Petersburg - Moscow, 2014. - 528 p.

14. Enzh Eric. SEO. Art of promotion of the websites; / Eric Enzh, Stefan Spencer, Rand Fishkin, Jesse Strikchiola – BHV-St. Petersburg - Moscow, 2014. - 668 p.

Рецензент: д.т.н., проф. Шворов С.А., Національний університет біоресурсів і природокористування

д.т.н., доц. Гунченко Ю.О., к.т.н., доц. Джулий В.Н.,
к.т.н., доц. Красильников С.Р., Солодеева Л.В., Чешун Д.В.

МЕТОДЫ АНАЛИЗУ И СИНТЕЗА РАЗРАБОТКИ WEB - ПРИЛОЖЕНИЙ

В статье предложен «ресурсный» метод организации контроллеров веб-приложения.

Архитектура программного обеспечения (компьютерной системы) представляет собой совокупность структур, которые состоят из программных элементов, снаружи видимых свойств этих элементов и взаимосвязей между ними. Архитектура - это неизменная глубинная структура веб-приложения. Ошибки, заложенные в архитектуру при ее проектировании, будут приводить к еще большим ошибкам в процессе ее реализации (кодировка). Самым перспективным на данный момент методом разработки веб-приложения есть метод разработки, основанный на парадигме "Модель-представление-контролер". Однако этот метод не лишен недостатков, самым существенным, из которых является отсутствие методов структурного синтеза программного кода уровня моделей, уровня представления и уровня контролеров. В связи с этим распространенным явлением стала неверная трактовка архитектуры, которая привела к появлению большого количества не унифицированного программного кода, что не удовлетворяют требованиям качества.

Для устранения рассмотренных недостатков предлагается модель архитектуры веб - приложения и метод структурного синтеза архитектуры, а также исходного кода уровня моделей, уровня представления и уровня контролеров. Синтез осуществляется методом группировки сущностей предметной области. Сущности веб-приложения необходимо представить специальными классами - моделями. Совокупность моделей представляет слой доступа к данным. В предложенном методе модельный слой охватывает меньшее множество функций, которые реализовываются, чем модельный слой в существующей архитектуре "Модель-представление-контролер" за счет введения сервисного слоя. На основе определений уровней абстракции становится возможным сформировать определение модуля веб-приложения. Модуль веб-приложения - совокупность представлений (шаблонов и видов), контролеров, моделей данных и базовых моделей, сервисов, конфигураций и тестов, отнесенных к определенной части предметной области приложения.

Предложенный "ресурсный" метод организации контролеров веб-приложения. Суть метода заключается в том, что необходимо проводить разработку уровня контролеров, базируясь не на моделях и представлениях, а на сущностях, над которыми планируется проводить операции. Такие сущности будут выступать в качестве ресурсов веб-дополнения.

Использование предлагаемой архитектуры и методик разработки веб-приложения позволит улучшить показатели качества и вместе с тем увеличить производительность разработки.

Ключевые слова: веб-приложения, контроллеры, модели, представления, программный код, метод, алгоритм, структурный синтез, интернет-технологии.

Ph.D. Gynchenko Y.A., Ph.D. Dzhulij V.M.,
Ph.D. Krasilnikov S.R., Solodeeva l.V., Chechun D.V.

METHODS AND ALGORITHMS WEB APPLICATION DEVELOPMENT

The"resource" method for organizing Web application controllers is proposed in the article.

The software architecture (of a computer system) is a structure' s set, consisting of the program elements, apparent properties of these elements and interactions with each other. Architecture is a depth structure of a web application. Any architecture mistakes admitted at the design stage, will lead to greater issues during the implementation (coding stage). The most promising development approach for now is the development pattern based on "Model-View-Controller" paradigm (MVC). However, this one is not without disadvantages. The most significant issue is the lack of methods for structural composition of software code on the model, view and controller levels. As result, there is an incorrect interpretation of

architecture, which led to the emergence of a huge amount of not unified software code that does not meet the quality requirements.

In order to solve these issues, we propose a web application's architecture model and a method for structural composition of architecture, as well as source code for model, view and controller levels. The composition is carried out by the method of grouping the subject domain entities. The web application entities are represented by special classes – models. The set of models belongs to data access layer. In accordance to the proposed method, the model layer covers a smaller set of implemented functions than the regular model layer in MVC architecture by introducing a service layer. It becomes possible to determine the web application module due to the abstraction level definitions. Web application module is a set of representations (view classes and templates), controllers, data models and base models, services, configurations, and tests assigned to a specific part of the application.

The "resource" method of organizing web application controllers is proposed. The general idea of the method is necessity of developing the controllers level, based not on views and models, but on the entities which will be processed. Such entities will serve as web application resources.

Applying the proposed architecture and web application development techniques will improve the quality and increase the development performance.

Keywords: web applications, controllers, models, views, program code, method, algorithm, structural composition, internet technologies.