При этом

$$b_1 = \frac{a_1}{4};\quad b_2 = \frac{1}{32}\left(8a_2 - 3a_1^2\right) - \frac{c_1}{4};\quad c_2 = \frac{7}{512}a_1^4 + \frac{3}{32}a_2a_1^2 - \frac{3}{8}a_2^2 + a_4 + \frac{3}{8}c_1^2;$$

$$c_3 = \frac{7}{4096}a_1^6 - \frac{1}{256}a_1^4a_2 - \frac{3}{64}a_1^2a_2^2 + \frac{1}{8}a_1^2a_4 - \frac{1}{2}a_2a_4 + \frac{1}{8}a_2^3 + a_6 +$$
$$+\left(\frac{1}{2}a_4 + \frac{7}{1024}a_1^4 + \frac{3}{64}a_1^2a_2 - \frac{3}{16}a_2^2\right)c_1 + \frac{1}{16}c_1^3;$$

$$c_4 = -\frac{39}{1048576}a_1^8 - \frac{5}{32768}a_1^6a_2 + \frac{1}{8192}a_1^4a_2^2 + \frac{3}{1024}a_1^4a_4 - \frac{1}{32}a_1^2a_2a_4 + \frac{3}{512}a_1^2a_2^3 +$$
$$+\frac{3}{32}a_1^2a_6 - \frac{3}{256}a_2^4 + \frac{1}{16}a_2^2a_4 - \frac{1}{4}a_2a_6 + a_8 +$$
$$+\left(\frac{7}{16384}a_1^6 - \frac{1}{1024}a_1^4a_2 - \frac{3}{256}a_1^2a_2^2 + \frac{1}{32}a_1^2a_4 - \frac{1}{8}a_2a_4 + \frac{1}{32}a_2^3 + \frac{1}{4}a_6\right)c_1 +$$
$$+\left(\frac{7}{8192}a_1^4 + \frac{3}{512}a_1^2a_2 - \frac{3}{128}a_2^2 + \frac{1}{16}a_4\right)c_1^2 + \frac{1}{256}c_1^4;$$

параметр c_1 остается свободным.

Для упрощения расчетов удобно выбирать $c_1 = 0$.

Исследование осуществлено в рамках договора БРФФИ № Ф21М-118 на выполнение НИР «Разработка новых методов нахождения корней алгебраических уравнений в символьном виде».

Заключение. В работе рассмотрены два случая декомпозиции алгебраического полинома восьмой степени на полиномы второй и четвертой степеней. В явном виде сформулированы необходимые и достаточные условия наличия исследуемых композиций, а также получены формулы вычисления коэффициентов полиномов, из которых образована композиция.

1. Трубников, Ю.В. Об условиях представимости полиномов четвертой и шестой степени в виде суперпозиции полиномов второй и третьей степени / Ю.В. Трубников, В.В. Юргелас // Веснік Віцебскага дзяржаўнага ўніверсітэта. – 2019. – № 1(102). – С. 17–24. Режим доступа: https://rep.vsu.by/handle/123456789/18080. – Дата доступа: 07.09.2022.

2. Астапов, И.С. Решение алгебраических уравнений третьей и четвертой степеней с помощью компьютерной алгебры / И.С. Астапов, Н.С. Астапов // Программная инженерия. – 2014. – № 10. – С. 33–42.

ПРИМЕНЕНИЕ ПАТТЕРНОВ ПРОЕКТИРОВАНИЯ ДЛЯ СОЗДАНИЯ TEST AUTOMATION FRAMEWORK (TAF)

Грицкевич Н.С.,
студент 3 курса ВГУ имени П.М. Машерова, г. Витебск, Республика Беларусь
*Научный руководитель – **Новый В.В.,** ст. преподаватель*

Ключевые слова. Тестирование, фрэймворк для автоматизации тестирования, паттерны проектирования для TAF.

Keywords. Testing, test automation framework, design patterns of TAF.

В настоящее время IT-технологии участвуют во многих процессах жизнедеятельности человека. Примером могут служить социальные сети, сайты для заказа еды, оплата коммунальных услуг и т. д. Но кто же проверяет, правильно ли работают эти все системы? Кто же выявляет дефекты в приложениях? Как быстро они могут это сделать? Этими нюансами занимаются Quality Assurance engeniers. Automation QA разрабатывает TAF для выявления дефектов на протяжении жизненного цикла приложения. В данной статье мы не будем рассматривать основы построения фреймворка, но рассмотрим, как можно,

используя паттерны проектирования, получить малозатратный по ресурсам и скорости выполнения авто-тестов TAF.

Цель работы – показать, какие паттерны чаще всего используются в TAF, и в каких случаях их нужно использовать.

Материал и методы. Материалом исследования являются основные паттерны, использующиеся в TAF [1]. Методы исследования – анализ, методы объектно-ориентированного программирования на языке java.

Результаты и их обсуждение. Рассмотрение основных паттернов TAF необходимо начать с паттерна Page Object, так как если попытаться найти какую-либо информацию в Интернете про паттерны TAF, то первое, что найдется – это Page Object.

Page object – шаблон проектирования, который используется при написаний авто-тестов представляет каждую html страницу как отдельный класс с элементами содержащимися на странице и методами взаимодействия с ними [2].

Основной задачей данного паттерна является повторное использование одной и той же страницы несколько раз, исключая дублирование кода. Не выгодно использовать данный паттерн, если приложение является одностраничным, что в наше время является очень редкой ситуацией, что и делает данный паттерн одним из «best practice». Приведем пример данного паттерна:

```
public class Page {
    private final WebElement applyButton = Selenide.$x("//button[./label]");
    public void clickOnApplyButton() {
    applyButton.click();
  }
    public boolean isApplyButtonDisplayed() {
    return applyButton.isDisplayed();
  }
    public boolean isApplyButtonEnabled() {
    return applyButton.isEnabled();
  }
    public boolean isApplyButtonDisabled() {
    return !applyButton.isEnabled();
  }
}
```

Singleton – паттерн проектирования, при котором объект данного класса возможно проинициализировать только один раз за все время компиляции. Для TAF этот паттерн также является одним из важных, так как в своем большинтсве пользователь взаимодействует со страницами с помощью Selenium webdriver, а создавать объект данного класса для каждой страницы будет неуместно и затратно. Вариаций реализаций singleton есть 6, и каждый нужно выбирать в зависимости от цели, которую вы хотите получить. Примером будет приведена самая распространенная версия Singleton в TAF в виде оболочки над Selenium Webdriver [3]:

```
public class SingletonDriver {
    private static SingletonDriver driver;
    private SingletonDriver() {}
    public static WebDriver getInstance() {
  if (driver == null) {
   switch (System.getProperty("browser", "chrome")) {
    case "chrome": {
     driver = WebDriverManager.chromedriver().create();
     break;
    }
    case "firefox": {
     driver = WebDriverManager.firefoxdriver().create();
     break;
```

```
      }
      case "opera": {
       driver = WebDriverManager.operadriver().create();
       break;
      }
      default: {
       driver = WebDriverManager.edgedriver().create();
      }
     }
    }
    driver.manage().window().maximize();
    return driver;
  }
  public static void tearDown() {
    driver.quit();
    driver = null;
  }
}
```

Таким образом, был описан пример классовой оболочки для selenium webdriver для получения объекта драйвера и для его закрытия.

Заключение. В работе рассмотрены два часто использующихся паттерна проектирования для создания гибкого TAF, а также примеры их использования. Данная работа может быть использована в качестве краткого руководства для начинающих test automation engineer.

1. П75 Паттерны объектно-ориентированного проектирования / Э. Гамма [и др.] – СПБ.: Питер, 2021. – 448 с.: ил. – (Серия «библиотека программиста»).

2. Page Object [Электронный ресурс]. – 2022. – Mode of access: https:\\martinfowler.com/bliki/PageObject.html. – Data of access: 10.09.2022.

3. Тестирование и оценка качества программного обеспечения [Электронный ресурс]: [учеб.-метод. комплекс] для студентов, обучающихся по спец. 1-31 03 07 Прикладная информатика / М-во образования Республики Беларусь, Учреждение образования "Витебский государственный университет имени П.М. Машерова", Математический фак., Каф. информатики и информационных технологий. – Электрон. текстовые дан. (1 файл: 11 Мб). – Витебск, 2014. – Режим доступа: lib.vsu.by.

РЕДАКТОР СХЕМ «СУЩНОСТЬ–СВЯЗЬ» КОНЦЕПТУАЛЬНЫХ МОДЕЛЕЙ

Демченко Н.Н.,
учащийся 3 курса Оршанского колледжа ВГУ имени П.М. Машерова,
г. Орша, Республика Беларусь
Научный руководитель – ***Юржиц С.Л.,*** *магистр образования, преподаватель*

Ключевые слова. ER-диаграмма, сущность, связь, ассоциация (элемент), базы данных, редактор диаграмм.

Keywords. ER-Diagram, entity, relationship, association (element), database, diagram editor.

Спустя десятки лет развития человечества в IT-сфере подходы и принципы создания программного продукта менялись неоднократно. С развитием технологий у человека появилось огромное поле для собственных идей. Если представить команду разработчиков, состоящую из большого количества человек, в процессе работы над программным продуктом могут быть недопонимания. Этого не стоит допускать в команде, ведь успех проекта зависит от эффективности ведения разработки. Для организации работы составляется документация на проект, одним из аспектов создания документации является концептуальная модель данных. Концептуальная модель данных лежит в основе систем управления и проектирования БД, и успех проектирования и разработки программного