УДК 685.34.024:004.4
Е.В. Чебанюк, В.И. Чупринка

АРХИТЕКТУРА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ПОСТРОЕНИЯ РАСКРОЙНЫХ СХЕМ ДЕТАЛЕЙ С ДЕКОРАТИВНЫМИ ЭЛЕМЕНТАМИ

Актуальность. Построение раскройных схем с нанесёнными на детали декоративными элементами позволяет расширить ассортимент изделий, которые выпускаютпредприятия обувной и кожгалантерейной ромышленности. В других отраслях нанесение фигурных отверстий позволяет спроектировать детали, готовые для соединенияв разные механизмы. Для представления архитектурных решений используются статические модели программного обеспечения, а именно UML диаграммы классов, пакетов и компонентов.

Постановка задачи: разработать архитектуру и статическую модель программного обеспечения построения раскройных схем деталей с декоративными элементами. Статическая модель программного обеспечения представляется диаграммой классов.

Требования к результату: разработанная архитектура и статическая модель программного обеспечения
(ПО), представленная диаграммой классов, должны удовлетворять следующим критериям:

- расширять программное обеспечение;
- позволять изменять конфигурации внешних контуров декоративных элементов;
- обеспечивать добавление и удаление декоративных элементов из коллекции, с затратой минимальных усилий разработчиков;
- лёгкая интеграция с существующими программными модулями для построения раскройных схем;
- минимизировать усилия разработчиков при сопровождении и тестировании программного обеспечения.

Настоящая работа является продолжением работ [1–4]. В работе [1] были рассмотрены алгоритмы расчёта координат внешних контуров декоративных элементов на основе их ключевых параметров и предложена методика сохранения информации о них. Все ключевые параметрыподелены на две категории. Первая категория – параметры,общие для всех декоративных элементов, а именно: Xp, Yp –координаты полюса декоративного элемента; PD – признак детали, на которую наносится декоративный элемент (название детали); $ПЕ$ – признак декоративного элемента (т.е.его название); φ – угол поворота декоративного элемента относительно его основного положения. Вторая – параметры, характерные для определённых декоративных элементов [1]. Например, для N-угольника это: Xp,Yp – координаты полюса N- угольника; N – количество сторон многоугольника; R –радиус описанного круга; φ– угол поворота N-угольника.

В работе [1] были рассмотрены такие декоративные элементы: прямоугольник, ромб, *N*-угольник, крест, окружность, эллипс, звезда и капля. В работе [2] было предложено решение следующих задач: генерирование групповых и одиночных декоративных элементов; разработка формата файлов для сохранения информации о деталях с декоративными элементами; описание алгоритмов обработки этих файлов. В работе [3] разработана алгебра, позволяющая сохранять информацию о статических моделях ПО, с возможностью группирования фрагментов диаграмм классов по паттернам проектирования. В работе [4] было предложено обоснование выбора паттерна проектирования для решения сформулированной задачи.

Рассмотрим основные процессы предметной области «Проектирование раскройных схем с нанесением декоративных элементов на детали». Есть коллекция декоративных элементов, заданных по характеристикам [1]. Для построения раскройной схемы с использованием деталей, на которые

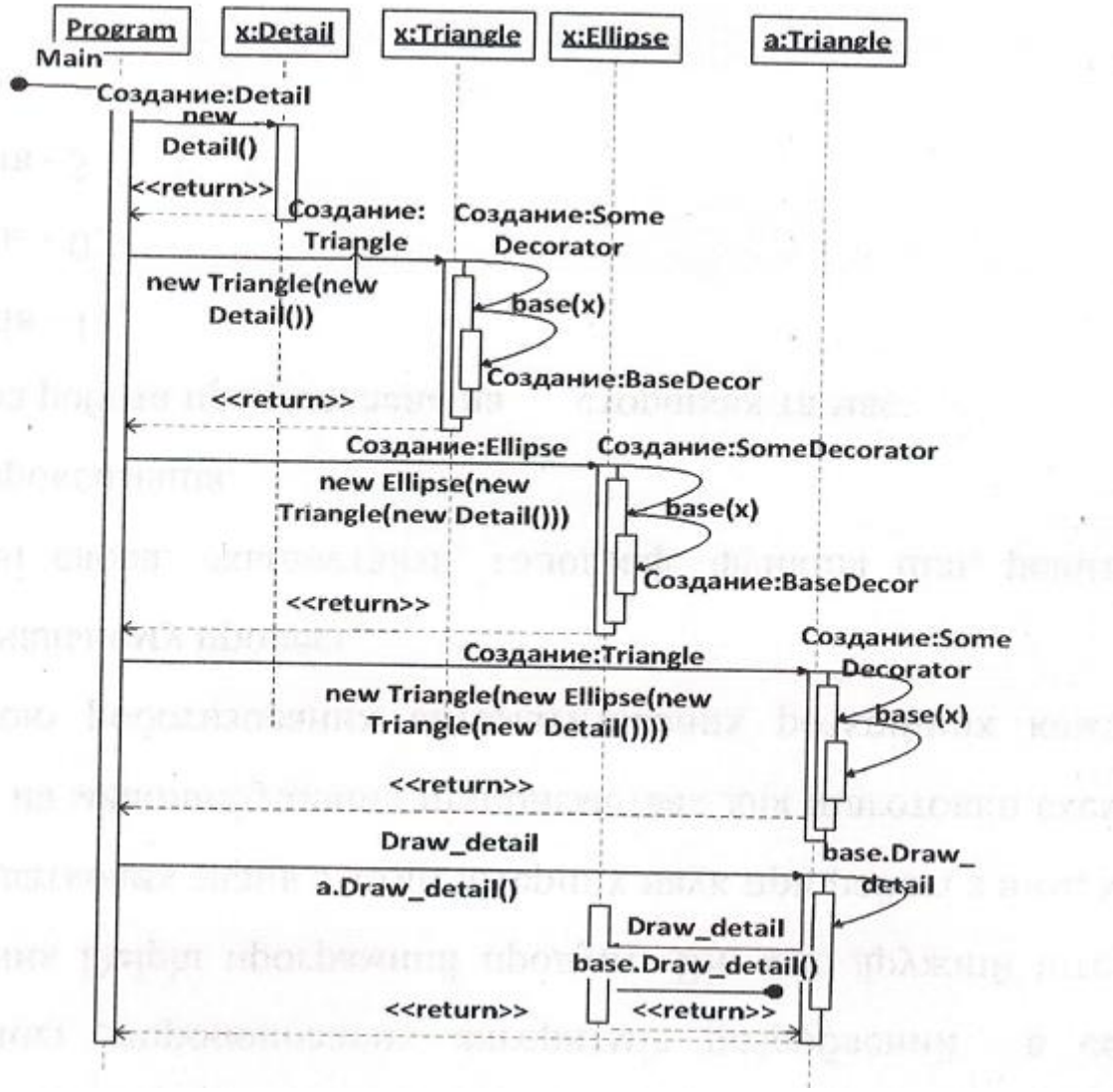

Рис. 1. Формализованное представление процесса нанесения декоративных элементов из коллекции на деталь

нанесены декоративные элементы, необходимо «декорировать» деталь выбранными элементами из этой коллекции или их комбинациями. Причём программное обеспечение должно позволять наносить или удалять групповые или одиночные декоративные элементы на деталь в произвольном порядке (наличие некоторого элемента из коллекции декоративных элементов не зависит от наличия других) для возможности гибко изменять ассортимент предприятия.

Использование паттернов проектирования позволяет получить следующие преимущества в проектировании ПО: применить устоявшиеся и проверенные архитектурные решения с возможностью гибкого расширения его функционала.

При проектировании диаграммы классов также сгруппируем её составляющие по паттернам проектирования с целью облегчить процесс создания сервисов и компонентов или пакетов, состоящих из программных модулей. Такой подход позволит свести к минимуму усилия разработчиков при сопровождении программного обеспечения. Диаграмма классов для решения этой задачи приведена на рисунке 2.

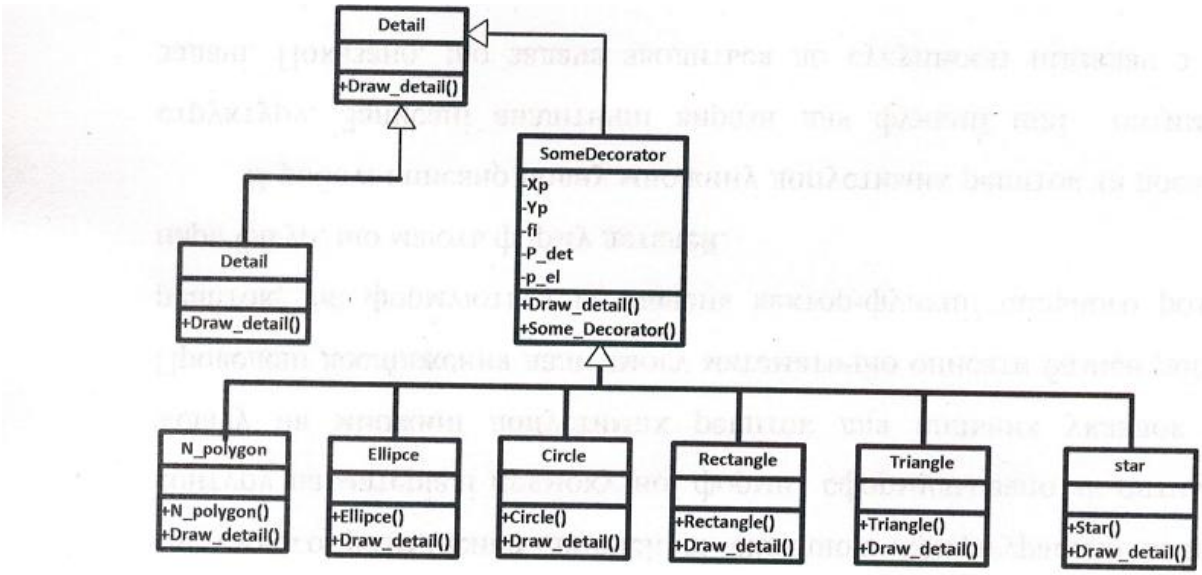

Рис. 2. Диаграмма классов для решения задачи нанесения декоративных элементов из коллекции на деталь

Представленная диаграмма классов позволяет использовать шесть видов декоративных элементов, а именно элементы, описанные в работе [1].

Свойства, характерные для всех декоративных элементов, инкапсулированы в класс SomeDecorator. При необходимости добавить декоративный элемент нужно унаследовать класс этого элемента от класса SomeDecorator. При использовании другой схемы именования классов необходимо использовать паттерн проектирования «адаптер». Тогда модель ПО будет содержать композицию составляющих этих паттернов проектирования.

Пример фрагмента кода на языке C#, который позволяет создать и прорисовать деталь раскройной схемы с нанесением на неё следующих декоративных элементов(рис. 1): треугольник, эллипс и квадрат.
BaseDecor a = new Triangle(new Ellipce(new Square(new Detail())));
a.Draw_detail().

Используя алгебру представления статических моделей ПО, предложенную в работе [3], сформируем аналитическое представление статической модели ПО, представленной диаграммой классов.
Для примера из коллекции декоративных элементов был выбран элемент «треугольник» (рис. 2). Описание функциональности и перегруженных методов остальных классов, представляющих другие элементы коллекции, выглядит подобным образом:

$$\begin{cases} \begin{cases} C^a(Base\_Decor) = B \\ C(Base\_Decor)^{abstract} = B(Base\_Decor)^{abstract} = \{\beta_0^{abstract} \mid \beta_0^{abstract} = Draw\_Detail\} \end{cases} \\ F(C(Detail)^{inh} = F(C^a(Base\_Decor)) \bigcup F(C(Detail)) \\ C(Detail)^{override} = \{\beta_0^{override} \mid \beta_0^{override} = Draw\_Detail\} \\ \begin{cases} F(C(SomeDecorator))^{inh} = F(C^a(Base\_Decoe) \bigcup (SomeDecorator)) \\ C(SomeDecorator)^{override} = B(SomeDecorator)^{override} = \{\beta_0^{override} \mid \beta_0^{override} = Draw\_Detail\} \end{cases} \\ \begin{cases} F(C(Triangle))^{inh} = F(C(triangle) \bigcup F(C(SomeDecorator)) \\ C(Triangle)^{override} = B(Triangle)^{override} = \{\beta_0^{override} \mid \beta_0^{override} = Draw\_Detail\} \end{cases} \end{cases}$$

Выводы: Программное обеспечение, предложенное в работе [2], реализующее алгоритмы сохранения информации о декоративных элементах с использованием ключевых параметров, предложенных в работе [1], ограничивало нанесение декоративных элементов, позволяя проектирование только групповых декоративных элементов. Это ограничение было вызвано тем, что необходимо было учитывать пересечение контуров декоративных элементов.

Предложенная в этой работе архитектура ПО для построения раскройных схем деталей с нанесёнными на них декоративными элементами позволит создавать программные продукты для моделирования новых изделий без этого недостатка. ПО, разработанное в соответствии с предложенной архитектурой, позволяет гибко модифицировать коллекции декоративных элементов и их составляющие. Применение такого ПО позволит сгенерировать множество дизайнерских решений, используя как одиночные, так и групповые декоративные элементы, нанесённые на деталь. нанесены декоративные элементы, необходимо «декорировать» деталь выбранными элементами из этой коллекции или их комбинациями. Причём программное обеспечение должно позволять наносить или удалять групповые или одиночные декоративные элементы на деталь в произвольном порядке (наличие некоторого элемента из коллекции декоративных

элементов не зависит от наличия других) для возможности гибко изменять ассортимент предприятия.

Библиографический список

1. Чупринка В.И., Чебанюк Е.В. Алгоритм сохранения информации о декоративных элементах на деталях
обуви // Техническое регулирование: базовая основа качества товаров и услуг: сб. науч. тр. –Шахты: ЮРГУЭС, 2009. –С. 70–3.
2. Чебанюк О.В., Чупринка В.І. Математичне та програмне забезпечення побудови розкрійних схем з деко-
ративними елементами // Вісник Східноукраїнського національного університету імені Володимира Даля. –
2010. – № 9(151). – С. 194–199.
3. Chebanyuk E. Algebra describing software static models. International journal Information technologies and
knowledge. – 2013. – № 1(Vol. 7.) – P. 83–93.
4. Чебанюк Е.В. Проектирование программного обеспечения для построения раскройных схем рулонных
материалов // Фундаментальные и прикладные исследования, разработка и применение высоких технологий в промышленности и экономике: XVI Междунар. науч.-практ.конф., 26–30 апр. 2013. – СПб., 2013. – № 2. – С. 168–171.