ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ (НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)»

Журнал практики

Студента ___Сорокин Илья Николаевич_____

(фамилия, имя, отчество)

Институт №3 «Системы управления, информатика и электроэнергетика»___

Кафедра № 319 «Системы интеллектуального мониторинга»_

Учебная группа__М3О-235Б-19____

Направление подготовки 09.03.01

(шифр)

«Информатика и вычислительная техника»

(название направления)

Вид практики _____*учебная*_____

(учебной, производственной, преддипломной или другой вид практики)

Руководитель практики от МАИ

Зеленова Марина Викторовна _____

(фамилия, имя, отчество) *(подпись)*

_____/_____/ “19” июля 2021 г.

(подпись студента) *(дата)*

Москва 2021

1.Место и сроки проведения практики

Сроки проведения практики:

-дата начала практики_____29 июня 2021 г._____

-дата окончания практики___19июля 2021 г._____

Наименование предприятия кафедра 319 «Системы интеллектуального мониторинга»

Название структурного подразделения (отдел, лаборатория)

_____ _____

2.Инструктаж по технике безопасности

1. Вводный инструктаж.

Проводится со следующими категориями слушателей:

- все сотрудники, принимаемые на работу;
- работники, прибывшие в организацию в командировку, и работники других организаций, которые заняты на определенных участках;
- для обучающихся в учебных заведениях, которые направлены в организацию на производственную практику;
- другие лица, занятые в производственной сфере работы предприятия.

Включает:

- инструктаж по технике безопасности на рабочем месте;
- инструктаж по охране труда;
- описание направления деятельности предприятия и отдела.

2. Первичный инструктаж на рабочем месте.

Должен проводиться перед тем, как сотрудник начнет самостоятельную работу. Его обязаны прослушать:

- все принятые на предприятие сотрудники, сюда относятся и те, кто заключил с организацией трудовой договор, работают на дому, а также по совместительству;
- сотрудники, переведенные с одного рабочего места на другое либо выполняющим данный вид работ впервые;
- командированные из других предприятий, временные работники, учащиеся образовательных учреждений, направленные на производственную практику, и другие сотрудники, чья работа связана с производственной деятельностью организации.

Включает:

- инсталляцию программного и аппаратного обеспечения для информационных и автоматизированных систем для анализа данных и машинного обучения;
- проверку состояния вычислительного оборудования;

- осуществление необходимых профилактических процедур вычислительного оборудования.

Шевченко Д.А. _____/ “29”июня 2021 г.
(подпись проводившего) *(дата проведения)*

3.Индивидуальное задание студенту

Предварительное определение темы и объема работ

1. Изучение различных паттернов проектирования.
2. Изучение порождающих паттернов.
3. Подробное изучение следующих паттернов: Factory method, Abstract factory, Builder

“29” июня 2021 г.

(дата проведения)

4.План выполнения индивидуального задания

План работ

1) Составление технического задания.

2) Определение паттернов.

3) Изучение видов паттернов.

4) Углубленное изучение трёх порождающих паттернов: Factory method, Abstract factory, Builder.

5) Написание программ-макетов с использованием каждого из методов.

Руководитель практики от МАИ: _Зеленова М.В.___/_____/

_____/_____/ “_29_” _июня__ 2021 г.
(подпись студента) *(дата)*

5.Отзыв руководителя практики от МАИ

За время прохождения практики Сорокин И. Н. зарекомендовал себя как ответственный и исполнительный практикант, показал высокий уровень теоретической подготовки, хорошее умение применить и использовать полученные знания для решения поставленных перед ним задач.

Программа практики выполнена полностью.

В целом работа практиканта Фамилия И.О. заслуживает оценки «Отлично».

Материалы, изложенные в отчете студента, полностью соответствуют индивидуальному заданию.

За время прохождения практики были сформированы следующие компетенции:

Шифр	**Компетенция**
ПК-1	Способность разрабатывать модели компонентов информационных систем, включая модели баз данных
ПК-3	Способность обосновывать реализуемые проектные решения, осуществлять постановку и выполнение экспериментов по проверке корректности и эффективности эт
ПК-7	Способность проверять техническое состояние вычислительного оборудования и осуществлять необходимые профилактические процедуры

И соответствующие им результаты освоения.

Руководитель практики от МАИ: _Зеленова М.В.___/_____/

(фамилия, имя, отчество) подпись)

«_19_»_июля_2021 г

6.Отчет студента о практике

В процессе прохождения практики мною были выполнены следующие задания:

1. Изучение различных паттернов проектирования.
2. Изучение порождающих паттернов.
3. Подробное изучение следующих паттернов: Factory method, Abstract factory, Builder
4. Написание программ-макетов каждого из трёх порождающих паттернов: Factory method, Abstract factory, Builder

Паттерн проектирования

Шаблон проектирования или **паттерн** (англ. design pattern) в разработке программного обеспечения — повторяемая архитектурная конструкция, представляющая собой решение проблемы проектирования в рамках некоторого часто возникающего контекста.

Обычно шаблон не является законченным образцом, который может быть прямо преобразован в код; это лишь пример решения задачи, который можно использовать в различных ситуациях. Объектно-ориентированные шаблоны показывают отношения и взаимодействия между классами или объектами, без определения того, какие конечные классы или объекты приложения будут использоваться.

«Низкоуровневые» шаблоны, учитывающие специфику конкретного языка программирования, называются идиомами. Это хорошие решения проектирования, характерные для конкретного языка или программной платформы, и потому не универсальные.

На наивысшем уровне существуют **архитектурные шаблоны**, они охватывают собой архитектуру всей программной системы.

Алгоритмы по своей сути также являются шаблонами, но не проектирования, а вычисления, так как решают вычислительные задачи.

Из чего состоит паттерн?

Описания паттернов обычно очень формальны и чаще всего состоят из таких пунктов:

- проблема, которую решает паттерн;
- мотивации к решению проблемы способом, который предлагает паттерн;
- структуры классов, составляющих решение;
- примера на одном из языков программирования;
- особенностей реализации в различных контекстах;
- связей с другими паттернами.

Такой формализм в описании позволил создать обширный каталог паттернов, проверив каждый из них на состоятельность.

Классификация паттернов по предназначению:

- Порождающие паттерны (Creational)
- Структурные паттерны (Structural)
- Поведенческие паттерны (Behavioral)

Описания видов паттернов

Порождающие паттерны отвечают за удобное и безопасное создание новых объектов, а некоторые даже позволяют создавать семейства объектов. Паттерн, порождающий классы, использует наследование, чтобы изменять наследуемый класс, а паттерн, порождающий объекты, делегирует инстанцирование другому объекту.

Структурные паттерны отвечают за построение удобных иерархий классов. Структурные шаблоны уровня *класса* используют наследование для составления композиций из интерфейсов и реализаций. Простой пример — использование множественного наследования для объединения нескольких классов в один. В результате получается класс, обладающий свойствами всех своих родителей. Особенно полезен этот паттерн, когда

нужно организовать совместную работу нескольких независимо разработанных библиотек.

Поведенческие паттерны решают задачи эффективного и безопасного взаимодействия между объектами программы. В поведенческих паттернах уровня класса используется наследование, чтобы определить поведение для различных классов. В поведенческих паттернах уровня объекта используется композиция. Некоторые из них описывают, как с помощью кооперации несколько равноправных объектов работают над заданием, которое они не могут выполнить по отдельности. Здесь важно то, как объекты получают информацию о существовании друг друга. Объекты-коллеги могут хранить ссылки друг на друга, но это усиливает степень связанности системы. При высокой связанности каждому объекту пришлось бы иметь информацию обо всех остальных. Некоторые из паттернов решают эту проблему.

Какие есть порождающие паттерны?

Структурные паттерны подразделяются на:

- Фабричный метод
- Абстрактная фабрика
- Строитель
- Прототип
- Singleton (одиночка)

Рассмотрим три из них: фабричный метод, абстрактная фабрика, строитель.

Фабричный метод

Фабричный метод — это порождающий паттерн проектирования, который определяет общий интерфейс для создания объектов в суперклассе, позволяя подклассам изменять тип создаваемых объектов.

Паттерн Фабричный метод предлагает создавать объекты не напрямую, используя оператор new, а через вызов особого фабричного метода.

На первый взгляд, это может показаться бессмысленным: мы просто переместили вызов конструктора из одного конца программы в другой. Но теперь вы сможете переопределить фабричный метод в подклассе, чтобы изменить тип создаваемого продукта.

Цель фабричного метода

Определяет интерфейс для создания объекта, но оставляет подклассам решение о том, на основании какого класса создавать объект. Фабричный метод позволяет классу делегировать создание подклассов.

Используется, когда:

- классу заранее неизвестно, объекты каких подклассов ему нужно создавать;
- класс спроектирован так, чтобы объекты, которые он создаёт, специфицировались подклассами;
- класс делегирует свои обязанности одному из нескольких вспомогательных подклассов, и планируется локализовать знание о том, какой класс принимает эти обязанности на себя.

Структура фабричного метода

На рисунке 1 представлена структура фабричного метода:

- Product — продукт

определяет интерфейс объектов, создаваемых абстрактным методом;

- ConcreteProduct — конкретный продукт

реализует интерфейс Product;

- Creator — создатель

объявляет фабричный метод, который возвращает объект типа Product. Может также содержать реализацию этого метода «по умолчанию»;

может вызывать фабричный метод для создания объекта типа Product;

- ConcreteCreator — конкретный создатель

переопределяет фабричный метод таким образом, чтобы он создавал и возвращал объект класса ConcreteProduct.

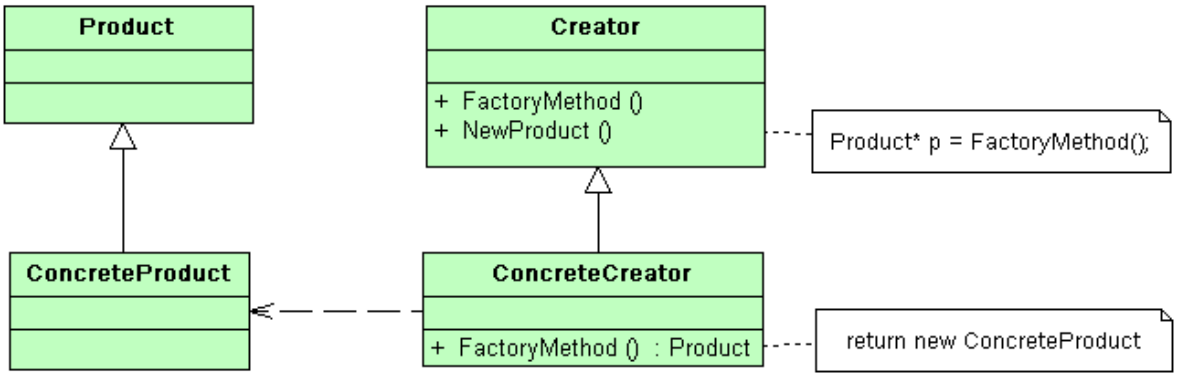

Рисунок 1 – Структура фабричного метода

Абстрактная фабрика

Абстрактная фабрика — это порождающий паттерн проектирования, который позволяет создавать семейства связанных объектов, не привязываясь к конкретным классам создаваемых объектов.

Назначение:

Предоставляет интерфейс для создания семейств взаимосвязанных или взаимозависимых объектов, не специфицируя их конкретных классов.

Используется, когда:

- Система не должна зависеть от того, как создаются, компонуются и представляются входящие в неё объекты;
- Входящие в семейство взаимосвязанные объекты должны использоваться вместе и вам необходимо обеспечить выполнение этого ограничения;
- Система должна конфигурироваться одним из семейств составляющих её объектов;

- Требуется предоставить библиотеку объектов, раскрывая только их интерфейсы, но не реализацию.

Структура абстрактной фабрики

На рисунке 2 представлена структура фабричного метода:

- Product — продукт

объявляют интерфейсы продуктов, которые связаны друг с другом по смыслу, но выполняют разные функции;

- AbstractProduct — абстрактный продукт

большой набор классов, которые относятся к различным абстрактным продуктам, но имеют одни и те же вариации;

- ConcreteFactory — конкретная фабрика

относятся каждая к своей вариации продуктов (Викторианский/Модерн) и реализуют методы абстрактной фабрики, позволяя создавать все продукты определённой вариации;

- AbstractFactory — абстрактная фабрика

объявляет методы создания различных абстрактных продуктов.

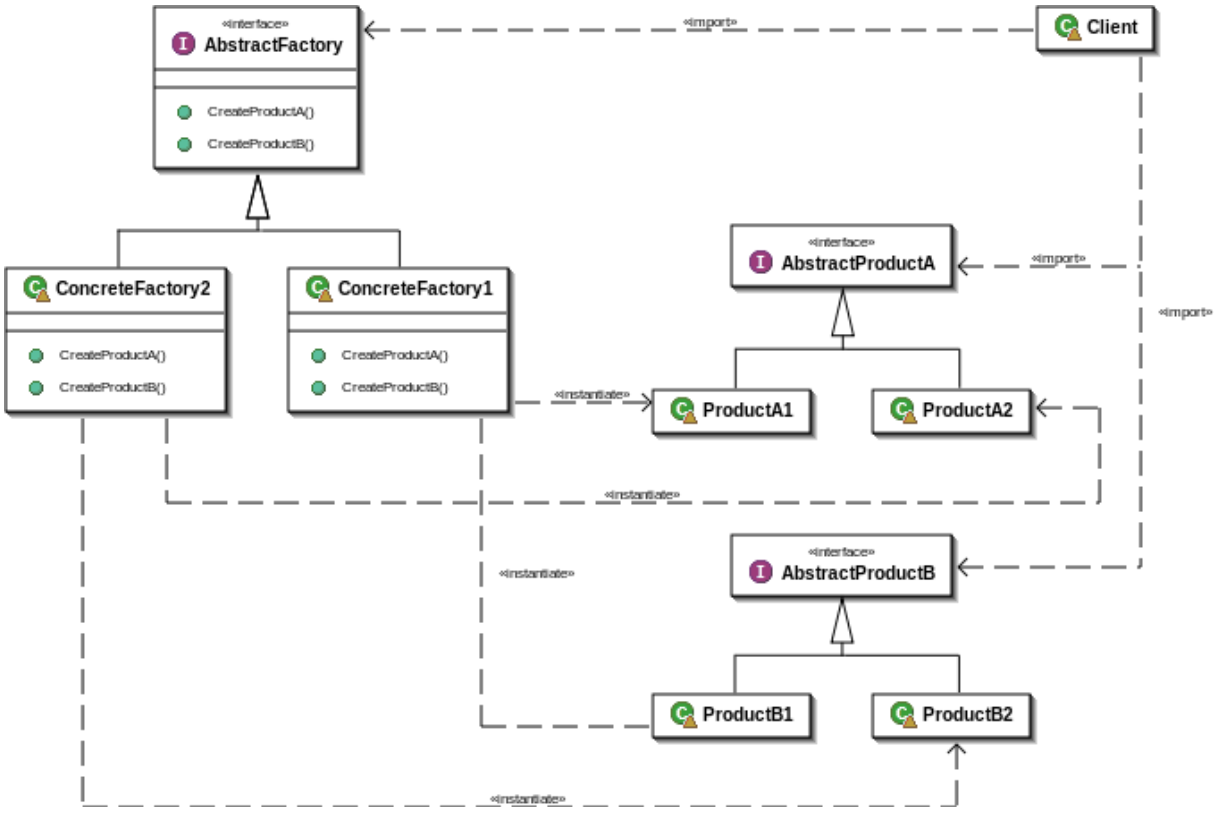

Рисунок 2 – Структура абстрактной фабрики

Строитель

Строитель — это порождающий паттерн проектирования, который позволяет создавать сложные объекты пошагово. Строитель даёт возможность использовать один и тот же код строительства для получения разных представлений объектов.

Назначение:

Отделяет конструирование сложного объекта от его представления так, что в результате одного и того же процесса конструирования могут получаться разные представления.

Используется, когда:

- Хочется избежать создания дополнительных конструкторов для использования опциональных параметров основного конструктора объекта;
- Необходимо создавать разные представления одного объекта;
- Есть необходимость собирать сложные составные объекты.

Структура абстрактной фабрики

На рисунке 2 представлена структура фабричного метода:

- Director - директор

определяет порядок вызова строительных шагов для производства той или иной конфигурации продуктов.

- Builder - интерфейс строителя

объявляет шаги конструирования продуктов, общие для всех видов строителей;

- ConcreteBuilder – конкретные строители

реализуют строительные шаги, каждый по-своему. Конкретные строители могут производить разнородные объекты, не имеющие общего интерфейса;

- Product - продукт

создаваемый объект. Продукты, сделанные разными строителями, не обязаны иметь общий интерфейс.

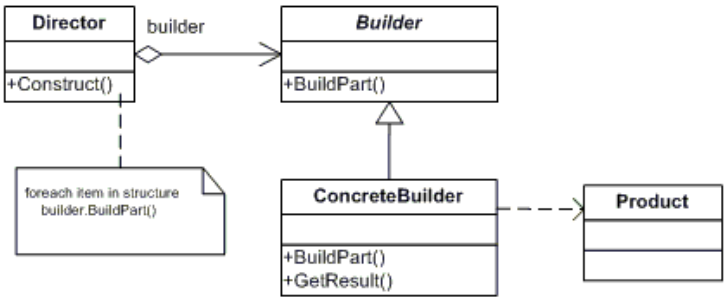

Рисунок 3 – Структура Строителя

ПРИЛОЖЕНИЕ. МОДЕЛЬ ВИНТОМОТОРНОЙ ГРУППЫ ТУ-95

Код программ – макетов каждого из разобранных в работе порождающих паттернов находится в архиве SorokinPracticeJournal.zip и представляется на съемном носителе.

Эта страница НЕ РАСПЕЧАТЫВАЕТСЯ, просто для сведения

Примечание:

1. На титульном листе после подписи студента должна ставиться дата окончания практики;
2. В п.№3 записывается формулировка темы задания на практику;
3. План выполнения индивидуального задания (п.№4) оформляется на отдельном листе, и после подписи студента должна ставиться дата начала практики;
4. В п.№5 желательно руководителем практики от предприятия завершать отзыв фразой:

   «Материалы, изложенные в отчёте студента, полностью (или не полностью) соответствуют индивидуальному заданию».
5. Отзыв руководителя практики от предприятия пишется на отдельном листе;
6. Отчет студента по практике пишется на отдельных листах и его объем устанавливается руководителем практики от МАИ.