**Практическая работа 5**

**Паттерн:** Chain of Responsibility (Цепочка Обязанностей)

**Пример:** Плагин Спам-фильтр

**Задача:**

Проверить является ли входящее письмо спамом

**Метод реализации:**

Письмо проходит по цепочке ряд обработчиков, которые подсчитывают параметр SpamScore. Т.е. на основе содержимого письма определяют можно ли его отнести к спаму. Все начинается с проверки на вредоносные вложения и вирусные ссылки, ссылки сна запрещенные/подвергнутые цензуре ресурсы. Потом мы проверяем сам текст сообщения по неким характерным признакам – по критичности – начиная с самых очевидно (наличие рекламных сообщений, навязчивое предложение услуг предложение).

(Пока не реализовано:

Проверка отправителя – сверяемся с черным списком отправителей

На последней стадии применяется анализатор пользовательских предпочтений на основе содержимого почтового ящика (анализ проведен заранее, у нас есть доступ к набору интересующих пользователя тем)

Пользователь так же может изменять настройки фильтрации. В зависимости от его выбора в цепочку добавляются различные обработчики. )

**Преимущества CoR при решении задачи:**

Мы можем остановить обработку на любом из обработчиков (при достижении определенного Spam Score письмо будет отправлено в папку Спам)

Цепочка зависит от выбора пользователя (динамически изменяема)

Легко масштабируема – в любой момент можно добавить сколько угодно новых обработчиков и изменения в коде будут минимальны

**Сравнение с шаблонами, применяющимися при решении схожих задач**

**Шаблон**: Список обработчиков (Chain of Responsibility pattern vs **List of Handlers**)

**Аргументы в пользу CoR:**

Цепочка организована так, что обработчик может:

• Выполнить некоторую работу перед запуском следующего обработчика, включая изменение входящего параметра для следующего обработчика;

• выполнить некоторую работу после запуска следующего обработчика, включая обработку или изменение выходных данных следующего обработчика, прежде чем он будет возвращен предыдущему обработчику;

• использовать информацию, сохраненную до запуска следующего обработчика, чтобы впоследствии повлиять на вывод.

Чтобы реализовать подобную функциональность без CoR (с помощью списка), потребовалось бы больше обратных вызовов и более сложный жизненный цикл обработчика.

**Др. шаблоны (сочетания шаблонов), которые иногда предлагают в качестве альтернативы CoR**

**(хотя все это сводится к вопросу List vs CoR)**

Список Команд (Chain of Responsibility vs **Commands in a queue / stack**)

Список Стратегий (Chain of responsibility pattern vs **List of Strategies**)

Цепочка Фильтров (**FilterChain** почти полный аналог CoR в Java)