МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский авиационный институт (национальный исследовательский университет)»

Институт № 6 «Аэрокосмический»

Кафедра 604 «Системный анализ и управление»

Реферат

На тему: «Исследование и разработка принципов построения инструментальных средств конфигурирования в плагинных системах»

По курсу: «Системный анализ»

Выполнил: Шаблий А.Д.
Аспирант группы: М8О-108А-23
Проверила: к.т.н. Дьячук А. К.

Москва 2024

Содержание

1 Введение

В современном мире важную роль в жизни общества занимают информационные технологии. В частности информационные технологии охватывают программные средства, применяемые для цифровых устройств, которые человек использует в повседневной жизни.

Отдельно выделяют класс программных средств - плагинные системы. Это способ организации приложения таким образом, что его конечный объем функционала характеризуется количеством установленных в него расширений - плагинов.

Каждый отдельный плагин включает в себя конечное множество функционала, который в свою очередь основан на требованиях. Именно характер реализованных требований, а так же их объем потребен заказчику. Все остальное - плагины, их взаимосвязи, язык программирования, на котором они реализованы, библиотеки, которые задействованы - все это скрыто от заказчика и зачастую его не интересует. Заказчику интересно, какие свои бытовые или бизнес потребности он сможет удовлетворить от применения программного средства.

На стадии проектирования приложения зачастую неизвестно, какие требования будут востребованы у заказчиков и определить структуру приложения невозможно. Кроме того, для разных заказчиков потребен разный функционал. При необходимости формирования поставки, включающей требуемый объем функционала, поставщик зачастую включает и тот функционал, который не востребован и не оплачен заказчиком, но без которого не быть поставлен востребованный. Это связано с существованием зависимостей у реализации.

Моя диссертационная работа посвящена поиску оптимальной структуры приложения, которая бы с одной стороны позволяла формировать поставки с минимальным числом невостребованного у заказчика функционала, а с другой сдерживала неконтролируемый рост кодовой базы, тем самым, сдерживала стоимость разработки и сопровождения проекта.

2 Описание системы

В качесте системы для исследования я обозначил структуру программного средства, предназначенного для интеграции в плагинную систему. Надсистемой является жизненный цикл ПО. Для выявления и описания подсистем мною проанализирована предметная область, сформулированы ограничения на компоненты системы и предложена формулировка ее описания в виде графовой модели. Модель состоит из следующих сущностей, к которым применены ограничения:

1. сущности классифицированы:
   (a) включающая конкретный перечень плагинов поставка;
   (b) пригодные для интеграции в плагинную систему плагины;
   (c) распределенные по плагинам файлы исходного кода;
   (d) функциональные зависимости между файлами исходного кода и плагинами;

(e) реализованные в файлах исходного кода требования.

2. один файл исходного кода может реализовывать одно или несколько требований;

3. одно требование может быть реализовано в одном или нескольких файлах исходного кода;

4. один файл исходного кода не может быть включен в несколько плагинов;

5. файлы исходного кода могут иметь зависимости друг на друга, в том числе и циклические;

6. плагины так же имеют зависимости друг на друга: если в первый плагин включены файлы, зависящие от файлов, расположенных во втором плагине, то первый плагин зависим от второго;

7. циклические зависимости между плагинами запрещены.

Принципиальная схема приведена на рисунке 1.

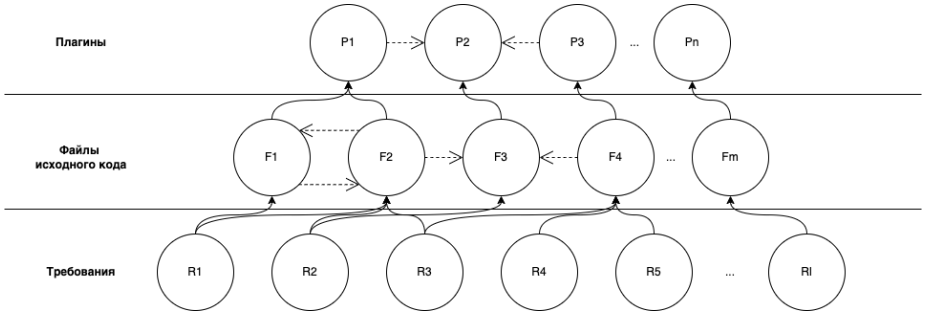

Рис. 1: Система с приведенными подсистемами

3 Аспекты внешней среды

Внешними факторами для системы являются исходный объем требований ТЗ R^* и требования текущего заказчика R^*_n. При формировании поставки, в нее включается реализация требования непотребных заказчику R^*_{un} в связи с наличием в проекте функциональных зависимостей. Поставляемый объем требований $R^*_d = R^*_n \cup R^*_{un}$. От $|R^*_{un}|$ зависит коэффициент бесполезности K_f. Этим коэффициентом характеризуется степень холостой работы подразделений постпродажного обслуживания сформированной поставки компанией производителем. $K_f = |R^*_{un}|/|R^*_d|$.

Например, заказчик запросил 10 требований, а поставлен был функционал, реализующий 15. 5 требований он не оплатил, но осуществлять их поддержку все равно необходимо. В данном случае $K_f = 5/15 = 1/3$. Почти 33% работы

персонала по постпродажному обслуживанию поставки программного средства заказчиком не оплачена.

Графическая интерпритация приведенных множеств приведена на рисунке 2.

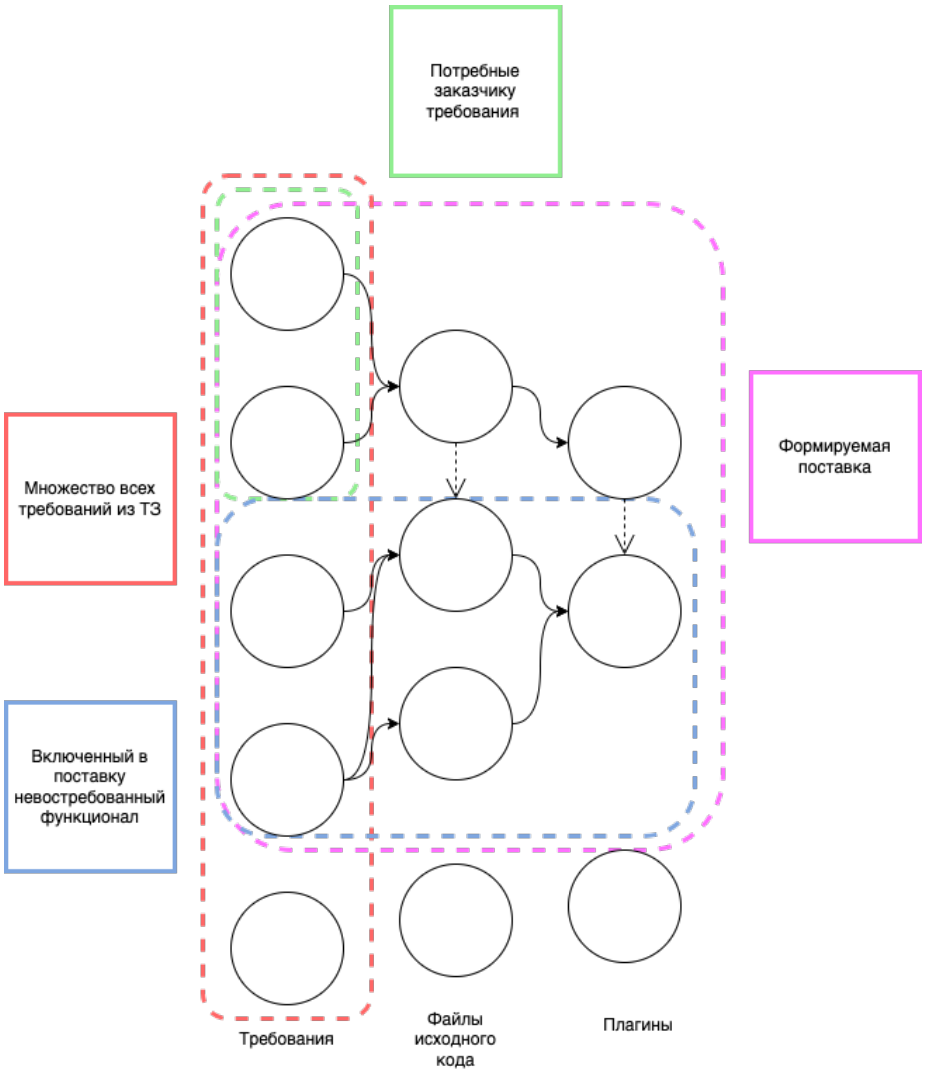

Рис. 2: Формирование поставки по потребным заказчику требованиям

С другой стороны, существуют затраты на разработку и поддержку программного средства. Чем больше его кодовая база V_c, тем дороже его разработка, отладка и управление его конфигурацией.

Подытоживая, от показателей K_f и V_c зависит рентабельность программного средства, а значит и конкурентоспособность бизнеса.

4 Выявление альтернатив системы

С целью уменьшения K_f видится применение схемы «1 требование - 1 файл исходного кода - 1 плагин». Но этот вариант при достаточно большом объеме требований приведет к неконтролируемому росту V_c в проекте ПО, следовательно, неконтролируемый рост затрат на разработку и поддержку. Чтобы их компенсировать производитель будет вынужден повышать цену на свое программное средство, что снижает конкурентоспособность бизнеса.

Другой альтернативой является схема «все требования - 1 плагин». В этом случае K_f будет слишком велик если заказчик заинтересован лишь в части возможностей всего продукта. Службы постпродажного обслуживания будут перегружены, что так же приводит к стоимости поставляемого решения и снижения конкурентоспособности бизнеса.

В своей диссертации я решаю задачу оптимальной декомпозиции функционала по плагинам с целью снижения K_f и контроля роста показателя V_c.

5 Описание модели

Попытки описания механизмов, позволяющих реализовать разделение функционала уже существуют в наше время. Примером является Equenox - реализация спецификации OSGI, которая лежит в основе IDE Eclipse.

Спецификация OSGI предполагает построение (синтез) системы из обособленных компонентов. Для разделения зависимостей на сторонние библиотеки и сервисы. Степень разделения и ее характер зависит от реализации. В IDE Eclipse она реализована для решения целей и задач стоящих перед самой IDE, однако предполагает ее расширение по правилам конкретного бизнеса.

В рамках диссертационного иссследования, следуя правилам спецификации OSGI, мною на языке программирования Java написано решение подзадачи изменения исходного графа с учетом разрешения циклических зависимостей между файлами исходного кода. В решении получены результаты экспериментов для оценки времени работы приложения от типа используемых коллекций, в которых хранится информация о графе:

1. вершины - файлы исходного кода и требования к ПО;
2. ребра - зависимости между файлами исходного кода и трассируемость требований к ПО.

Работа алгоритма заключается в итеративном поиске циклов в графе с последующим слиянием входящем в цикл вершин до тех пор, пока в графе не будут отсутствовать циклы.

Схема разработанного решения приведена на рисунке 3.

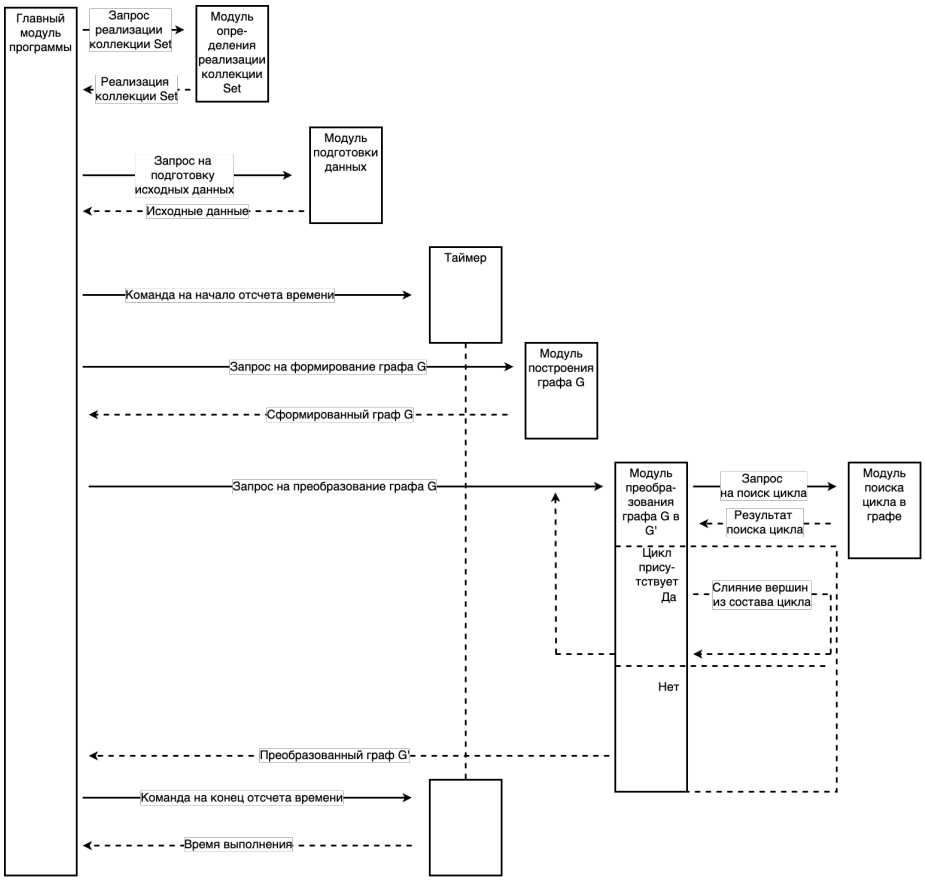

Рис. 3: Схема разработанного решения

В дальнейшем предполагается доработка модели для поиска оптимального распределения файлов исходного кода по плагинам.

Для выявления лучшего решения сформулирован векторный критерий:

1. возможность динамеческого ценнобразования поставки C_d;
2. V_c;
3. K_f;

C_d является определяющим фактором. Именно благодара ему достигается динамическое изменение K_f при статических R^* и V_c.

$$C_d = \begin{cases} 1 & \text{если } \{R^{'}|R_d^*\} \quad \exists R^{'} \in R_{un}^* \\ 0 & \text{если } \{R^{'}|R_d^*\} \quad \nexists R^{'} \in R_{un}^* \end{cases}$$

V_c определяет общий объем кодовой базы, поддержку которого необходимо осуществлять. Чем больше объем, тем более дорогой является поддержка.

$$V_c = \begin{cases} 1 & \text{если } N_p = 1 \\ 2 & \text{если } 1 < N_p < N_f \\ 3 & \text{если } N_p = N_f \end{cases}$$

K_f не статичен и может быть уникален для каждой из поставок.

Так, вектора показателей эффективности для сформулированных альтернати в предлагаемого решения следующие:

1. альтернатива по схеме «1 требование - 1 файл исходного кода - 1 плагин»:
   - $C_d = 1$;
   - $V_c = 3$;
   - $K_f = 0$.
2. альтернатива по схеме «все требования - 1 плагин»:
   - $C_d = 0$;
   - $V_c = 1$;
   - $K_f \approx 1$.
3. предлагаемое решение:
   - $C_d = 1$;
   - $V_c = 2$;
   - $0 < K_f < 1$.

В предлагаемом решении C_d = 1, что обеспечивает главное условие - динамическое изменение K_f при статических R^* и V_c. Несмотря на увеличение V_c по сравнению со схемой «все требования - 1 плагин», V_c не увеличивается на столько как в схеме «1 требование - 1 файл исходного кода - 1 плагин». Применяя различные алгоритмы оптимизации, приследуется цель выявления такой декомпозиции, которая обеспечивала бы минимизацию значения K_f. Предполагается использовать следующие алгоритмы оптимизации:

1. генетический алгоритм;
2. PageRank;
3. решающий лес.

6 Заключение

Полученные мною результаты применяются для различных приложений и решений выполненных как набор плагинов для интеграции в соответствующие им плагинные системы.

Так, на конференции в МГТУ им. Баумана я в своем докладе «Исследование правил построения конфигуратора ARINC 653 спецификации в IDE Eclipse» показал результаты применения сформулированной модели для решения задачи путем

разбиения файлов исходного кода по плагинам по частотам вызова функций. В качестве результата продемонстрировал снижение невостребованного функционала до 10% при различных запросах состава требований.

На конференции в Воронеже в ВУНЦ ВВС «ВВА имени профессора Н.Е. Жуковского и Ю.А. Гагарина» в своем докладе «Реализация инструментального средства конфигурирования компонентов БРЭО» показал актуальность применения плагинных систем для решения задач в авиационной сфере.

Подготовил выступление на конференции, организуемой Союзом Машиностроителей России. В качестве материала выступления мною предоставлено средство интеллектуального конфигурирования системы определения состояния воздушного судна. В этой работе я сформировал среду конфигурирования, которая независимо подключается к базе данных и заполняет ее конфигурационными параметрами и их значениями с целью дальнейшей их обработки непосредственно в самой системе.

Подготовлена публикация ВАК, в которой описан алгоритм разрешения циклических зависимостей графовой модели трассируемости требований к ПО на файлы исходного кода. Работа передана в редакцию ИПМ им. М.В.Келдыша РАН.