**Проектирование архитектуры программных систем**

**Отчёт о лабораторной работе №2**

**Выполнил Акимов Андрей, 21ПИ-1**

**Использованные приёмы программирования**

**Включение** — это метод переиспользования компонентов, при котором внешний компонент является клиентом для внутреннего компонента, содержащего необходимую функциональность. Внешний компонент получает указатели на интерфейсы внутреннего компонента и использует их для взаимодействия. При необходимости внешний компонент может заново реализовать интерфейс, поддерживаемый внутренним, передавая последнему вызовы этого интерфейса. Также внешний компонент может специализировать этот интерфейс, добавляя свой код перед вызовом внутреннего компонента и после него.

Агрегирование — это особый случай включения, при котором внешний компонент передаёт указатель на интерфейс внутреннего компонента клиенту. Внешний компонент не реализует интерфейс заново для передачи вызовов внутреннему компоненту. Если компонент агрегирован, его делегирующий компонент передает вызовы внешнему компоненту. Если компонент не агрегируется, его делегирующий компонент передает вызовы неделегирующему компоненту.

Осуществление приёмов программирования

Для осуществления включения требуется выполнить следующие шаги:

Создать в новом компоненте поле для хранения ссылки на уже существующую реализацию нужного интерфейса и новую виртуальную таблицу для этого интерфейса.

Запросить реализацию нужного интерфейса у существующего компонента при инициализации нового компонента с помощью метода QueryComponent() интерфейсной шины и сохранить эту реализацию в соответствующее поле нового компонента.

Реализовать методы виртуальной таблицы интерфейса, направляя их вызовы на ранее сохраненную реализацию интерфейса.

Для осуществления агрегирования требуется выполнить следующие шаги:

Добавить в компонент поле типа IEcoUnknown \*, в котором будет храниться интерфейс агрегируемого компонента. Через этот интерфейс можно будет запросить другие интерфейсы, которые реализованы во внутреннем компоненте.

Также необходимо добавить в компонент поле с реализацией неделегирующего собственного интерфейса IEcoUnknown

Запросить у нужного компонента с помощью метода QueryComponent() интерфейсной шины интерфейс IEcoUnknown и передать ссылку на себя при инициализации компонента. Это будет явное намерение на агрегирование. После этих действий необходимо полученный указатель сохранить в соответствующем поле компонента.

В методе QueryInterface() компонента направлять запросы на нужные интерфейсы, реализованные внутренним компонентом, с использованием сохраненного указателя.

Дополнительно:

Если запросы к компонентам не удается выполнить внутри компонента, их следует перенаправлять на внешний компонент.

Если компонент не агрегируется, все запросы должны быть перенаправлены на неделегирующий интерфейс.

При агрегировании необходимо проверять условия, чтобы избежать возможной рекурсии.

Порядок включения и агрегирования компонентов калькулятора

Реализация интерфейса Y содержится в компонентах A и B, а реализация интерфейса Y – в компонентах E и D. Что касается компонента C, то здесь возникает конфликт, так как он имеет один и тот же идентификатор, что и компонент A.

Также стоит отметить, что агрегирование возможно только с компонентом B.

Процесс включения компонента, необходимого для работы с интерфейсом X:  
Первым шагом мы пытаемся агрегировать компонент B. Однако, если агрегирование компонента B не удается, мы переходим на второй шаг и обращаемся к компоненту A. Если и этот компонент оказывается недоступным, то работа с интерфейсом X становится невозможной, тогда все методы будут возвращать 0.

Процесс включения компонента, необходимого для работы с интерфейсом Y:

Первый шаг - включение интерфейс Y из компонента D. Если включение компонента D не удается, мы переходим к следующему шагу - обращаемся к компоненту E для доступа к интерфейсу Y. Если и этот компонент оказывается недоступным, то работа с интерфейсом Y также становится невозможной, тогда все методы будут возвращать 0.

Важно отметить, что компонент E также агрегирует компонент B, что позволяет запросить интерфейс X через него. Однако, поскольку интерфейс X не будет функционировать без компонента B, такой случай не рассматривается.

Результат работы программы

Проверка выполнения функций компонентов прошла успешно. Ответ, полученный с помощью них, совпадает с стандартно выполненными операциями.