# РАБОТА №6 ПРАКТИЧЕСКАЯ. ОБРАТНОЕ ПРОЕКТИРОВАНИЕ АЛГОРИТМА

## 1. ЦЕЛЬ И ЗАДАЧИ РАБОТЫ

Целью работы является получение практических выполнения обратного проектирования.

## 2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Прямым проектированием (Forward engineering) называется процесс преобразования модели в код путем отображения на некоторый язык реализации. Процесс прямого проектирования приводит к потере информации, поскольку написанные на языке UML модели семантически богаче любого из существующих объектноориентированных языков. Фактически именно это различие и является основной причиной, по которой мы, помимо кода, нуждаемся и в моделях. Некоторые структурные свойства системы, такие как кооперации, или ее поведенческие особенности, например взаимодействия, могут быть легко визуализированы в UML, но в чистом коде наглядность теряется.

Обратным проектированием (Reverse engineering) называется процесс преобразования в модель кода, записанного на каком-либо языке программирования.

В результате этого процесса вы получаете огромный объем информации, часть которой находится на более низком уровне детализации, чем необходимо для построения полезных моделей. В то же время обратное проектирование никогда не бывает полным. Как уже упоминалось, прямое проектирование ведет к потере информации, так что полностью восстановить модель на основе кода не удастся, если только инструментальные средства не включали в комментариях к исходному тексту информацию, выходящую за пределы семантики языка реализации. Пример, представленный на рис. 3.3, был создан с помощью обратного проектирования библиотеки классов языка Java.

Процесс обратного проектирования делится на два этапа: анализ и генерацию модели.

На первом этапе производятся все подготовительные операции по анализу текста программы на отсутствие синтаксических ошибок. Второй этап – преобразование кода в модель.

Все операции выполняются независимо, что дает больший маневр для разработчика, который, например, хочет провести только синтаксический разбор теста, без генерации модели.

Соответственно при отсутствии ошибок в файле можно приступить к генерации модели. В целях оптимизации времени генерации в предусмотрено три способа проведения обратного проектирования, каждый из которых может охватить и превосходно выполнить определенный сегмент работ:

* FirstLook – приближенная пробежка по телу программы.  Detailed Analysis – детальный анализ проекта.
* RoundTrip – комбинация двух вышеперечисленных способов. Позволяет безболезненно строить и перестраивать разрабатываемые приложения по принципу круговой разработки.

Нашей целью будет получение графической модели из класса на языке программирования. Обратите внимание на комментарии. Каждая строка снабжена комментарием. Смысл обратного проектирования состоит не только в том, чтобы корректно нарисовать модель, но и для правильного описания спецификации каждой составляющей класса. За основу программы возьмем следующий класс:

|  |
| --- |
| //It's main class class string public: char \*string; //Structure's pointer int buffer[100]; //Temporary buffer char name[10]={«Massiv»}; //Name of data int a; //Integer  int b; //Integer void string(void); //constructor  void ~string(void); //destructor char StringCopy(char \*, //Buffer char \*, //sourcel char \*); //source2 private:  int tmp\_a; int tmp\_b; |

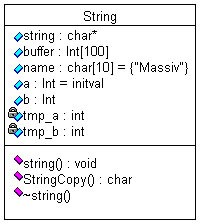


Рис.6.1 Класс, полученный в результате обратного проектирования

## 3. ПОРЯДОК ВЫПОЛНЕНИЯ И ЗАДАНИЯ ДЛЯ РАБОТЫ

Заданием для работы является исходный код, содержащий описание классов. Классы должны содержать операции и их реализацию.

Выполнение должно включать следующие этапы.

1. Анализ кода и выделение состав классов.
2. Отображение классов в среде моделирования. Для каждого класса в отчѐте по работе должно быть сделано пояснение – на основе какого программного объекта он выявлен.
3. Выявление переменных классов и отображение их в среде моделирования. Для каждой переменной класса в отчѐте по лабораторной работе должно быть сделано пояснение.
4. Выявление операций классов. Для каждого класса необходимо выявить операции и сделано пояснение о свойствах операции, еѐ параметрах и уровне в иерархии наследования.

## 4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что из себя представляет обратное проектирование?
2. Для чего выполняется обратное проектирование?
3. Что из себя представляет результат обратного проектирования?
4. Какие модели можно построить в результате обратного проектирования?
5. Существуют ли инструментальные средства для обратного проектирования ?