# **2. Проектирование классов**

Проектирование — это процесс определения архитектуры, компонентов, интерфейсов и других характеристик системы или её части. Результатом проектирования является целостная совокупность моделей, свойств или характеристик, описанных в форме, пригодной для реализации системы. Поэтому следует установить взаимосвязи между ключевыми абстракциями, найденными в предметной области.

На основе объектно-ориентированного анализа была создана контекстная диаграмма классов из которой можно выделить:

* базовый класс «Software» (программное обеспечение), в котором были выделены следующие поля: название, версия, стоимость и размер занимаемого места программы;
* классы-наследники базового класса: «ProgrammingSystems» (системы программирования), который имеет поле «язык/среда программирования», «SystemSoftware» (системные программы) и «ApplicationSoftware» (прикладные программы) с полями «специализация».
* классы «OperatingSystem» (операционная система), «Utility» (утилита) и «DeviceDrivers» (драйвера) наследуются от класса «SystemSoftware», они имеют поля «обеспечение работы (всего/программ/устройств)». От класса «ApplicationSoftware» наследуются классы «GeneralPurposePrograms» (программы общего назначения) с полем «виды» и «SpecialPurposePrograms» (программы специального назначения), у которого есть поле «сфера применения».

Для хранения указателей на интерфейс был спроектирован обобщенный контейнерный класс и от него производный шаблонный класс. Этот класс нужен для более удобной полиморфной обработки данных. Он реализует статический вектор.

В завершение проектирования классов, становиться возможным составление диаграммы классов. Диаграмма классов - это структурная диаграмма, демонстрирующая общую структуру иерархии классов системы и взаимосвязей между ними.

# **3. Логическая структура программы**

Логическая структура программы представлена на рис. 2.

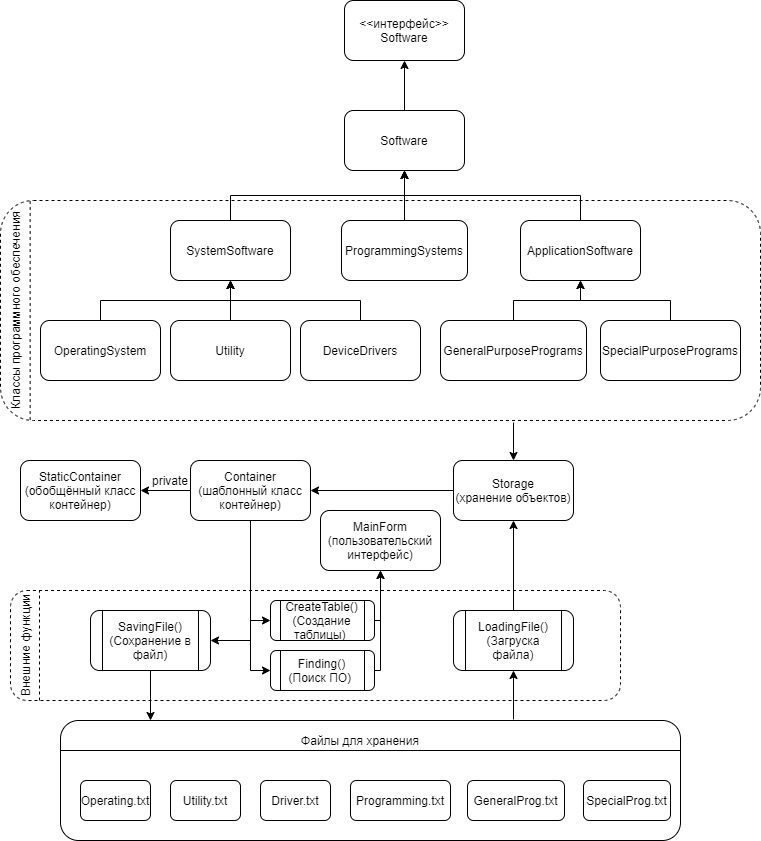


Рис. 2. Логическая структура программы.

Информация о каждом объекте хранится в файлах формата .txt. Внешняя функция LoadFromFile() загружает информацию об объектах из файлов, после чего создаётся таблица с этими данными в полях, которые помещаются в динамический вектор(Storage). С помощью шаблонного контейнера связываются объекты. Из пользовательского интерфейса происходит взаимодействие логических модулей и функций, так же пользователь может добавлять новые объекты, сохранять их в файл и осуществлять поиск по модели.