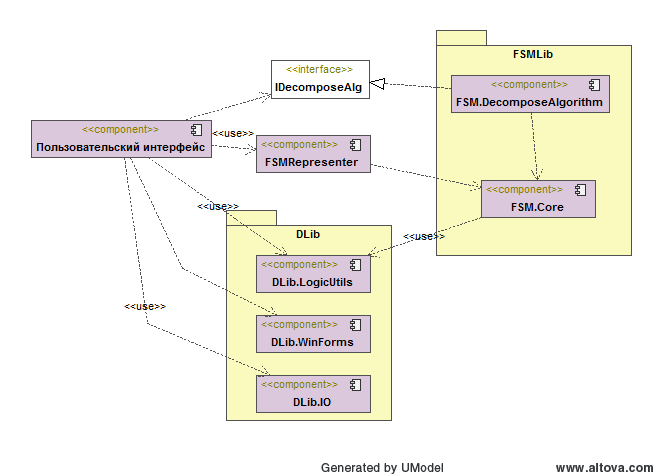
Общая структура программного продукта.

Для того чтобы учесть требование наличия возможности использования создаваемого программного продукта как части алгоритма анализа сложных систем, конечный программный продукт должен представлять собой полностью законченную библиотеку, позволяющую решать установленные в технических требованиях задачи.

Таким образом, программный продукт, получаемый в результате выполнения квалификационной работы, должен состоять из двух основных частей:

* программная библиотека, позволяющая создавать и инициализировать исходный вероятностный автомат, производить его декомпозицию и моделировать работы результирующей сети;
* программное приложение с графическим пользовательским интерфейсом, позволяющее использовать все заявленные функции описанной выше библиотеки.

На диаграмме представлена общая структура программы.



Программный комплекс состоит из следующих основных модулей:

* **FSM.Core** – содержит все примитивы, связанные с предметной областью теории конечных автоматов, включая определения вероятностного автомата и вероятностной сети автомата;
* **FSM.DecomposeAlgorithm** – содержит реализацию алгоритма декомпозиции вероятностного автомата;
* **IDecomposeAlg** – интерфейс для алгоритма декомпозиции вероятностного автомата, необходим для возможности дальнейшего расширения функционала разрабатываемой библиотеки за счёт увеличения методов декомпозиции;
* **FSMRepresenter** – данный модуль позволяет формировать различное представление данных из предметной области, например в виде таблиц или диаграмм;
* **Пользоватеский интерфейс**;
* **DLib** – утилитарная библиотека, используемая всеми модулями системы; содержит вспомогательные классы общего назначения;

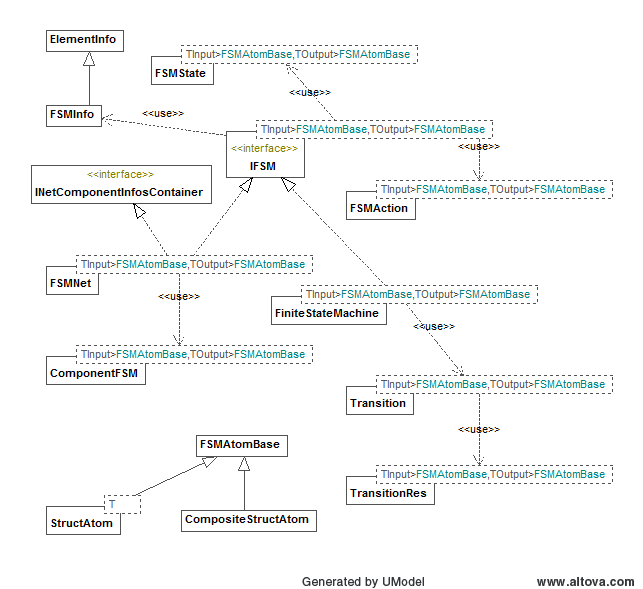
Рассмотрим подробнее модель предметной области.

Библиотека для работы с конечными автоматами.

При проектировании библиотеки необходимо учитывать тот факт, что в общем случае природа входного и выходного алфавитов неизвестен и определяется только в момент использования разрабатываемой библиотеки. Т.е. в качестве входного алфавита автомат может принимать символы, целые числа, различные структуры, определённые пользователем библиотеки и т.п. Для того, чтобы учесть данный аспект, было принято решение использовать механизм шаблонных типов при проектировании библиотеки.

К достоинствам данного подхода можно отнести использования строгой проверки типов при компилировании программы, что позволяет избежать определённых типов ошибок в процессе написания программного кода. При этом стоит отметить, что использование шаблонной типизации накладывает определённые ограничения на выбор языка программирования (см. технологический раздел).

На рисунке представлена принципиальная схема разрабатываемой библиотеки.



Основные сущности, используемые в библиотеке:

**FSMAtomBase** – базовая сущность для входных и выходных символов автомата. Реализует некоторые утилитарные интерфейсы, а так же логику идентификации, сравнения и копирования символов.

**StructAtom** – обёртка для простых типов входных и выходных символов автомата (целых чисел, строк и т.п.).

**CompositeStructAtom** – обёртка для составных типов входных и выходных символов автомата.

**IFSM** – интерфейс автоматной сущности. Подразумевает наличие имени, входного и выходного алфавитов, множества состояний, способности обрабатывать входные символы. Для определения сущностей данного типа необходимо указать шаблонные типы входных и выходных типов.

**FiniteStateMachine** – вероятностный конечный автомат, реализует интерфейс **IFSM**.

**FSMNet** – сеть вероятностных автоматов (подавтоматов). Реализует интерфейс **IFSM**. Содержит коллекцию подавтоматов.

**ComponentFSM** – подавтомат сети вероятностных автоматов.

**FSMState** – сущность состояния автомата, содержит информацию обо всех переходах из данного состояния.

**FSMAction** – сущность действия, совершаемого при срабатывании перехода.

**Transition** – сущность перехода из заданного состояния при воздействии определённого входного символа. Содержит коллекцию возможных исходов (**TransitionRes**).

**TransitionRes** – сущность исхода перехода (**Transition**). Содержит результирующее состояние, выходной символ и вероятность данного исхода.