Министерство образования и науки Российской Федерации
федеральное государственное бюджетное образовательное
учреждение высшего образования
«Алтайский государственный технический университет
им. И.И. Ползунова»

Факультет (институт) Информационных технологий
Кафедра Прикладная математика

Курсовой проект защищен с оценкой _____
Преподаватель_____ С. М. Старолетов
подпись

«___» _____ 2017 г.

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОМУ ПРОЕКТУ

Проектирование расширяемой библиотеки видеоэффектов, переходов и обработки слайдов

по дисциплине Архитектурное проектирование и паттерны программирования

КП 09.03.04.27.000 ПЗ

Студент группы ПИ-42 И. С. Хышов
подпись и.о., фамилия

Преподаватель доцент, к. ф.-м. н. С. М. Старолетов
(должность, учёное звание) подпись и.о., фамилия

Барнаул 2017

Задание

Учебная дисциплина: Архитектурное проектирование и паттерны программирования

ФИО студента: Хышов Иван Сергеевич

Группа: ПИ-42

Тема курсового проекта: Проектирование расширяемой библиотеки видеоэффектов, переходов и обработки слайдов

Этапы разработки курсового проекта и сроки их выполнения:

1. Изучение необходимой учебной и научно-технической литературы (01.10.2017 – 30.10.2017);
2. Разработка приложения (30.10.2017 – 20.11.2017);
3. Оформление отчета о проделанной работе (20.11.2017 – 20.12.2017);
4. Сдача работы руководителю и защита работы (20.12.2017 – 28.12.2017);

Дата выдачи задания: 01.10.2017

Срок защиты: 28.12.2017

Руководитель: доцент С. М. Старолетов

					КП 09.03.04.27.000 ПЗ			
Изм.	Лист	№ Докум.	Подпись	Дата				
Разраб.		Хышов И. С.			Проектирование расширяемой библиотеки видеоэффектов, переходов и обработки слайдов	Лит.	Лист	Листов
Пров.		Старолетов С. М.				У	2	30
						АлтГТУ, ФИТ гр. ПИ-42		
Н. Контроллер		Старолетов С. М.						
Утв.		Кантор С.А						

Содержание

Введение

На данный момент, в разнообразных отраслях, связанных с обработкой динамических изображений, будь то видеоиндустрия или же простая обработка слайдов для нужд собственного пользования, нередко приходится сталкиваться с приданием продукту требуемого вида, независимо от того, состоит этот продукт из одного или нескольких элементов. Как правило, под приданием требуемого вида можно понимать, к примеру, как изменение внешних характеристик отдельных компонентов (слайдов, видеоклипов, и т. д.), так и создание взаимодействий между несколькими компонентами.

Что характерно, потребности в наборе возможностей для композиции могут различаться, поэтому вышеупомянутый набор должен отвечать требованиям расширяемости и дополняемости, что не должно требовать перестройки всего инструментария полностью.

1 Анализ проблемы и существующих методов решения

1.1 Обзор предметной области

Видеоэффект, в терминах монтажа – совокупность техник, каким-либо образом преобразующих характеристики исходного видеоизображения к какому-либо виду или корректирующих их. Как правило, они, в основном, используются для визуализации условий, которые невозможно либо затруднительно снять в обычных условиях, а также для улучшения характеристик изображения. Наиболее распространенными из них являются такие эффекты, как фильтры цветокоррекции, шумоподавления, наложения других исходных изображений, имитационные фильтры, эффекты деформации и масштабирования, и другие эффекты.

Если говорить о цифровом нелинейном монтаже, то в качестве вышеупомянутых техник рассматриваются совокупности программно-аппаратных инструкций, преобразующих изображение, доступное в дискретизированном виде. На рисунке 1.1 можно увидеть примеры различных видеоэффектов.

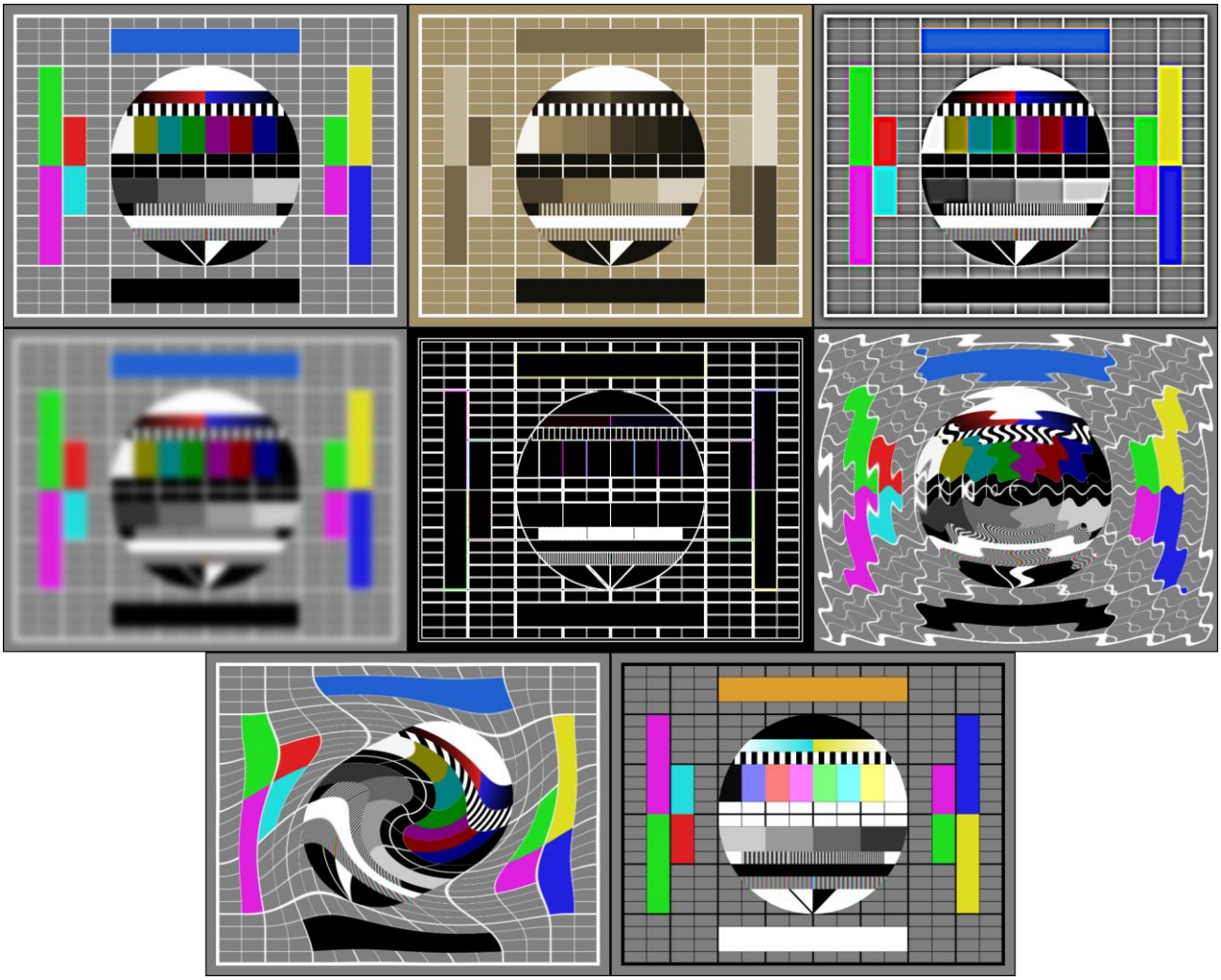

Рисунок 1.1 – Примеры видеоэффектов

Рассмотрим сферу применения видеоэффектов [5].

В различных сферах деятельности при публикации того или иного видеоматериала порой возникает потребность в изменении некоторых характеристик исходной визуальной составляющей для того, чтобы определенным образом повлиять на восприятие зрителем информации, содержащейся в используемом видеоизображении. Это может быть выделение определенных компонентов в информации, показывающейся при демонстрации видеоизображения, улучшение качества исходного изображения, полученного с дефектами определенной значимости, для улучшения восприятия содержащейся в ней информации, применение техник,

нацеленных на поддержание целостности и согласованности задуманной композиции и так далее. В некоторых случаях, как правило, в художественных или рекреационных целях, изображение изменяется определенным образом для достижения произвольных эффектов, которые способны воздействовать на психоэмоциональное состояние зрителя.

Перейдем к тонкостям реализации процесса монтажа и применения видеоэффектов к изображению. В качестве предметной области рассмотрим цифровой нелинейный монтаж, под которым понимается внос видеоматериала в компьютер, монтаж и применение эффектов, а затем – формирование итогового видеопроекта.

Если говорить о цифровом нелинейном монтаже, то следует упомянуть, что, в основном, в нем, в классическом понимании, используется одна или несколько дорожек, которые также можно охарактеризовать как слои, на которых располагаются исходные статические или динамические видеоизображения (также известные, как клипы), полностью или частично, в разных временных позициях. Клипы состоят из множества последовательно расположенных кадров одного и того же размера, которые могут быть как подготовленными заранее, так и генерированными в процессе отрисовки. В зависимости от используемых форматов, плотность кадров в одну секунду может различаться, однако, чем больше частота кадров, тем качественнее выглядит изображение. Каждый кадр представляет собой статичное изображение, в цифровом виде имеющее дискретную структуру (состоит из точек, в которых, в зависимости от формата, определены различные данные о цветах, например, красного, зеленого и синего для модели RGB) с ограниченным размером.

Как показано на рисунке 1.2, при отображении кадра, при отрисовке очередной точки используется точка того клипа, который находится на дорожке выше остальных, в той же позиции. В том случае, когда невозможно отрисовать точку клипа, используется фон. Как правило, это сплошной цвет, обычно – черный. Такая техника зачастую называется многослойным монтажом.

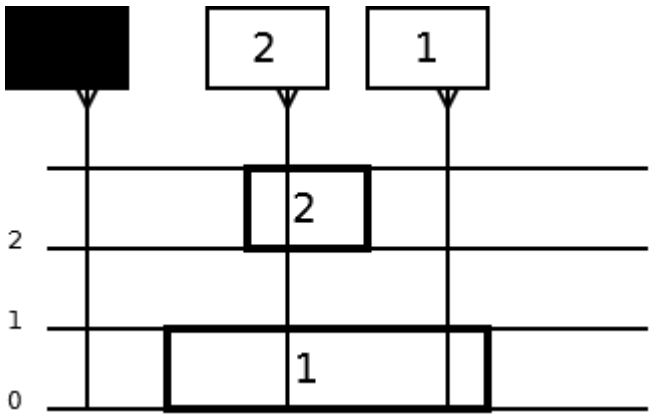

Рисунок 1.2 – Принцип выбора кадра для отрисовки при многослойном монтаже

При использовании видеоэффектов для преобразования исходного изображения, состоящего из множества последовательно расположенных кадров, работа производится с каждым отдельным его кадром по определенному алгоритму, однако в некоторых видеоэффектах могут использоваться и другие кадры клипа, причем не только смежные. В некоторых случаях в видеоэффектах может использоваться фоновое изображение (сплошной фон или кадр из клипа на нижестоящей или вышестоящей дорожке). Примером таких случаев можно назвать использование фильтра «Chroma key», заменяющего области, закрашенные определенным цветом (в классическом случае – синим или зеленым), соответствующими областями фонового изображения.

Видеоэффекты могут затрагивать как цветовую составляющую кадра, так и его геометрическую составляющую. К примеру, кадр может быть перемещен, масштабирован или повернут относительно рабочего пространства, может быть изменена его форма и так далее. Следует заметить, что при таких операциях часть кадра может выходить за границы рабочего пространства, а, следовательно – не отрисована.

Отдельной группой видеоэффектов стоит упомянуть монтажные переходы, как

способы соединения соседних клипов на одной дорожке. В простейшем случае, при переходе с одного клипа на другой происходит резкая смена кадра. Однако, возможны и более сложные случаи переходов, самыми распространёнными из которых являются затемнение, наплыв и вытеснение. Использование монтажных переходов широко распространено при создании презентаций, состоящих из слайдов, представляющих собой статичные или же динамические изображения. На рисунке 1.3 показан пример таких монтажных переходов, как наплыв и вытеснение.

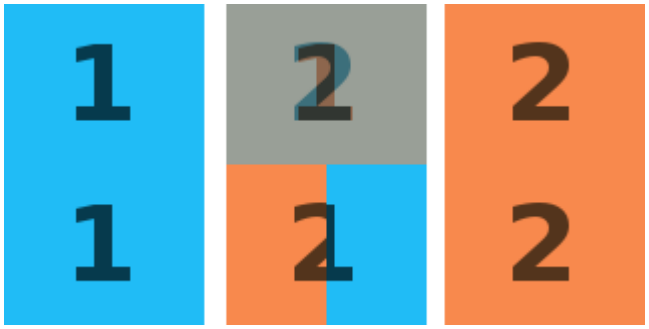

Рисунок 1.3 – Примеры монтажных переходов между клипами

В основном, эффекты монтажного перехода имеют приоритет ниже, чем у фильтров, преобразующих конкретный клип. Однако, при определенных условиях, возможно переопределить приоритет операций, к примеру, если фильтры применяются к клипу, к которому ранее был применен эффект перехода.

Видеоэффекты могут использоваться как поодиночке, так и в комбинации, причем в последнем случае процесс обработки происходит так: на вход первого фильтра подается исходный кадр, затем полученный обработанный кадр поступает на вход следующего фильтра, и так далее, пока на выходе последнего фильтра не будет получен результирующий кадр. Следует заметить, что на характеристики результирующего кадра особое влияние оказывает последовательность применяемых фильтров.

В некоторых случаях, влияние видеоэффектов на изображение может

динамически развиваться с течением времени. В таких случаях применяются различные техники, одну из которых представляют собой механизм ключевых кадров. При их использовании для определенного клипа задаются точки на временном промежутке, на которых устанавливаются определенные параметры используемых видеоэффектов, а при отрисовке результирующих кадров для используемых эффектов вычисляются промежуточные параметры с учетом значений, установленных для упомянутых ранее точек, при использовании определенных методов вычисления промежуточных значений, включающих методы интерполяции. Пример можно рассмотреть на рисунке 1.4.

Рисунок 1.4 – Пример видеоэффекта, изменяющегося с течением времени

Теперь следует рассмотреть проблемы реализации наборов видеоэффектов и переходов для программных средств, используемых для цифровой обработки видеоизображений.

Несмотря на то, что во многих существующих на сегодняшний день средствах, использующихся для монтажа видеоизображений присутствует определенный набор доступных для применения видов видеоэффектов, в некоторых случаях возникает потребность в использовании более специализированных и/или более сложных в реализации видов, которую не всегда удается удовлетворить при помощи использования только лишь штатных средств, а потому при разработке программного обеспечения, используемого для видеомонтажа требуется учитывать при проектировании его архитектуры возможность расширения набора доступных для использования видеоэффектов без необходимости перепроектировки всего программного продукта, которое, в идеале не только должно быть поддерживаемо со стороны разработчиков вышеупомянутых программных средств, но и не должно быть затруднено для сторонних разработчиков, желающих расширить функционал

разрабатываемых средств. В некоторых программных продуктах, к примеру, в некоторых видеоредакторах, присутствует возможность расширения набора видеоэффектов, отраженная в архитектуре программного продукта, которая может быть достигнута различными способами. К примеру, в программном продукте такая возможность может достигаться посредством использования подключаемых модулей, разрабатываемых как динамически подключаемые библиотеки, которые, как правило, разрабатываются с использованием тех же инструментариев, что и используются в рассматриваемом программном продукте, или набора скриптов, записанных на внутреннем или общедоступном языке. В любом случае, вложение расширяемости в архитектуру разрабатываемого программного продукта требует проектирования в нем достаточно функционального API (программного интерфейса приложения), посредством которого, к примеру, через его вызовы, должна производиться работа с определяемыми извне программными инструкциями, определяющими тот или иной эффект. Также при проектировании программных средств следует учитывать то, что особенности реализации библиотек видеоэффектов не должны зависеть от особенностей тех или иных форматов исходных данных, а посему должны обрабатывать исходные данные в общем порядке, ограничиваясь лишь внутренними особенностями рассматриваемого программного обеспечения.

1.2 Анализ существующих решений

Как правило, библиотеки видеоэффектов и переходов для монтажа зачастую бывают встроены в состав различных видеоредакторов и проигрывателей, как проприетарных, так и свободно распространяемых, однако, они также могут существовать в виде инструментариев для разработки. Рассмотрим некоторые из них.

MLT Multimedia Framework

Несмотря на то, что данный продукт [2] является не библиотекой, встроенной в какой-либо редактор, или самим инструментом для непосредственного монтажа, а лишь инструментарием для аудио- или видеомонтажа, предназначенным для интеграции в предназначенное для монтажа, воспроизведения и тому подобное программное обеспечение, этот свободно распространяемый кроссплатформенный инструментарий с модульной структурой имеет в себе обширный набор средств, включая поддержку большого числа форматов ввода/вывода медиа, оптимизированную библиотеку аудио-/видеофильтров и переходов, возможность подключения дополнительных модулей и так далее.

Frei0r

Свободный кроссплатформенный инструментарий [3] для использования простых фильтров, переходов для видеомонтажа, рассчитанных на использование при помощи параметризованных вызовов. Данный инструментарий не старается представить собой полноценный API для создания фильтров, однако он может использоваться с более высокоуровневыми API для расширения своей функциональности. Одна из главных особенностей этого инструментария – его легковесность, достигаемая за счет использования оптимизированных алгоритмов обработки и сведения зависимостей от сторонних библиотек к минимуму. В настоящий момент насчитывает в своем составе более 100 модулей.

OpenFX

OpenFX, также известный как OFX Image Effect Plug-in API [4] – это открытый стандарт для разработки подключаемых модулей для композиции и визуальных эффектов для двумерных изображений, которых позволяет подключаемым модулям работать с любым программным продуктом, использующим данный стандарт. Подключаемые модули разрабатываются как динамически разделяемые объекты, а API определяет некоторые входные точки, которые должны быть реализованы в подключаемом модуле. Каждый подключаемый модуль описывается списком параметров и поддерживаемыми входами и выходами. На данный момент, при помощи данного инструментария разработано большое количество подключаемых модулей для визуальных эффектов, а также поддержка данного стандарта включена во многие программные продукты, как свободно распространяемые, так и коммерческие.

2 Описание программного обеспечения

2.1 Структура реализованного приложения

Проектирование реализованного программного обеспечения можно разделить на несколько этапов:

Первый этап направлен на изучение предметной области, а также выбор языка программирования для реализации и используемых для этого библиотек. В результате данного этапа был выбран язык C++ с использованием инструментария Qt, однако упомянутый инструментарий был выбран только для реализации пользовательского интерфейса, тогда как основная часть проектируемого приложения была реализована с использованием штатных средств C++.

Второй этап заключается в проектировании системы и выделении в ней классов и интерфейсов, причем данный процесс осуществлялся с учетом используемых паттернов проектирования, отобранных для возможности расширения приложения в дальнейшем. В этот этап также входит подбор примеров возможного расширения для минимальной реализации. В случае данной предметной области, в качестве примеров возможного расширения были выбраны примеры фильтров и переходов. Например, из видов переходов был выбран наплыв, из видов фильтров были выбраны инвертирование и размытие.

В третий этап входят кодирование и отладка приложения.

2.2 Применяемые паттерны

При разработке приложения были использованы следующие структурные шаблоны[1] (они же – паттерны) для упрощения проектирования и дальнейшего расширения реализуемого программного продукта:

Синглтон

Паттерн «Синглтон» используется здесь для определения класса таким образом, чтобы объект, принадлежащий этому классу, существовал за все время выполнения приложения в единственном экземпляре и имел глобальную точку доступа. Здесь используется для классов MediaProject и EffectsLib, определяющих объекты медиапроекта (совокупность параметров отрисовки, определений исходных компонентов и их композиции) и библиотеки видеоэффектов и переходов соответственно, структура которых показана на рисунке 2.1.

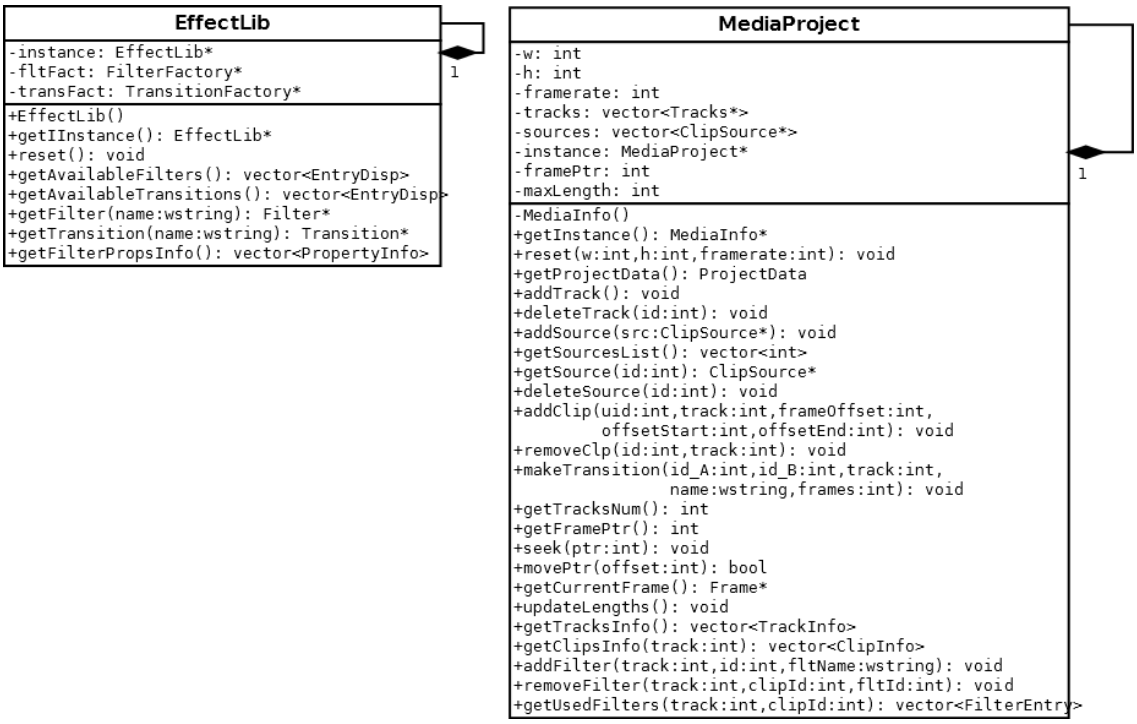

Рисунок 2.1 – Диаграмма классов для паттерна «Синглтон»

Фасад

Структурный паттерн «Фасад» используется для упрощения взаимодействия со сложной подсистемой классов извне. Эффект достигается за счет сведения взаимодействия с множеством объектов через вызовы к одному объекту, передающему вызовы другим объектам, что позволяет производить внутренние изменения без ведома пользователя. Используется в классе MediaProject, определяющем объект медиапроекта (см. рисунок 2.2). В этом классе предусмотрены методы для управления исходными данными (объекты класса ClipSource и наследуемых от него), а также дорожками (объекты класса Track), содержащими расставленные на них клипы (объекты класса Clip и наследуемых от него) без необходимости работать с объектами упомянутых ранее классов напрямую, что позволяет при дальнейшем развитии производить изменения в классах без необходимости изменений части, предназначенной для взаимодействий с пользователем, к примеру, интерфейса. Также данный паттерн реализован в объекте библиотеки видеоэффектов (класс EffectLib), при использовании которого пропадает необходимость в обращении к фабрикам видеоэффектов и переходов (объекты классов EffectFactory и TransitionFactory соответственно) напрямую (см. рисунок 2.3).

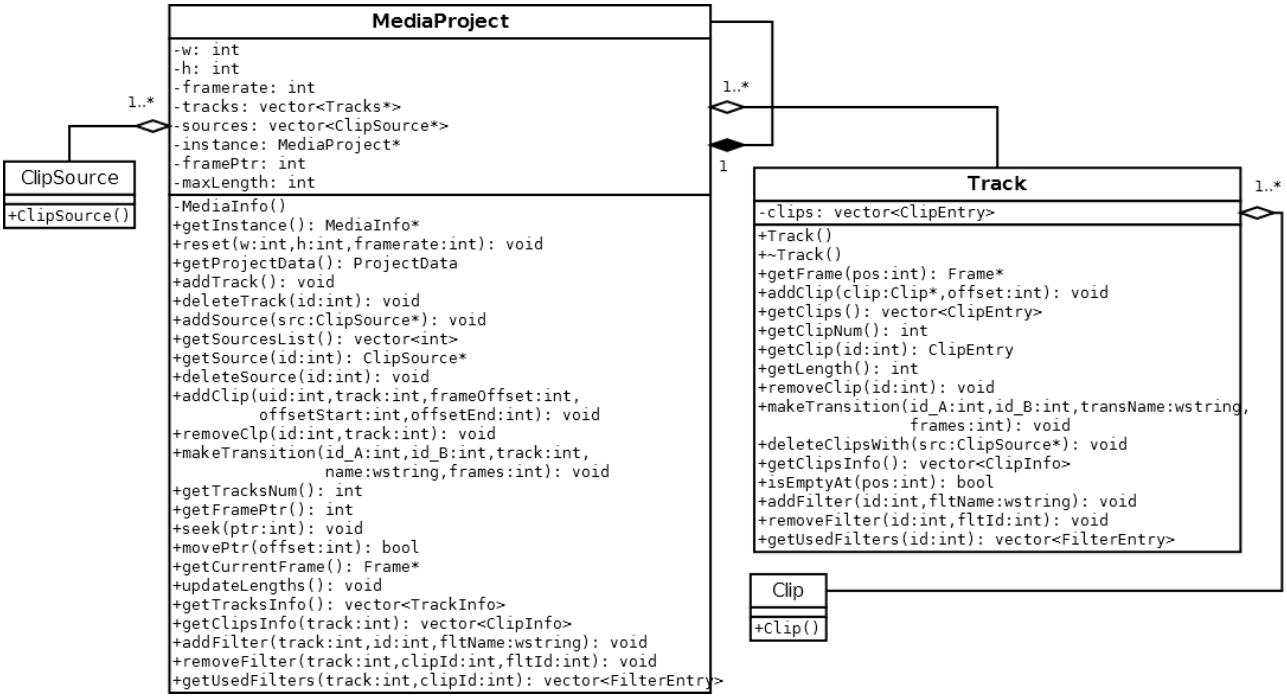

Рисунок 2.2 – Диаграмма классов для паттерна «Фасад» (класс MediaProject)

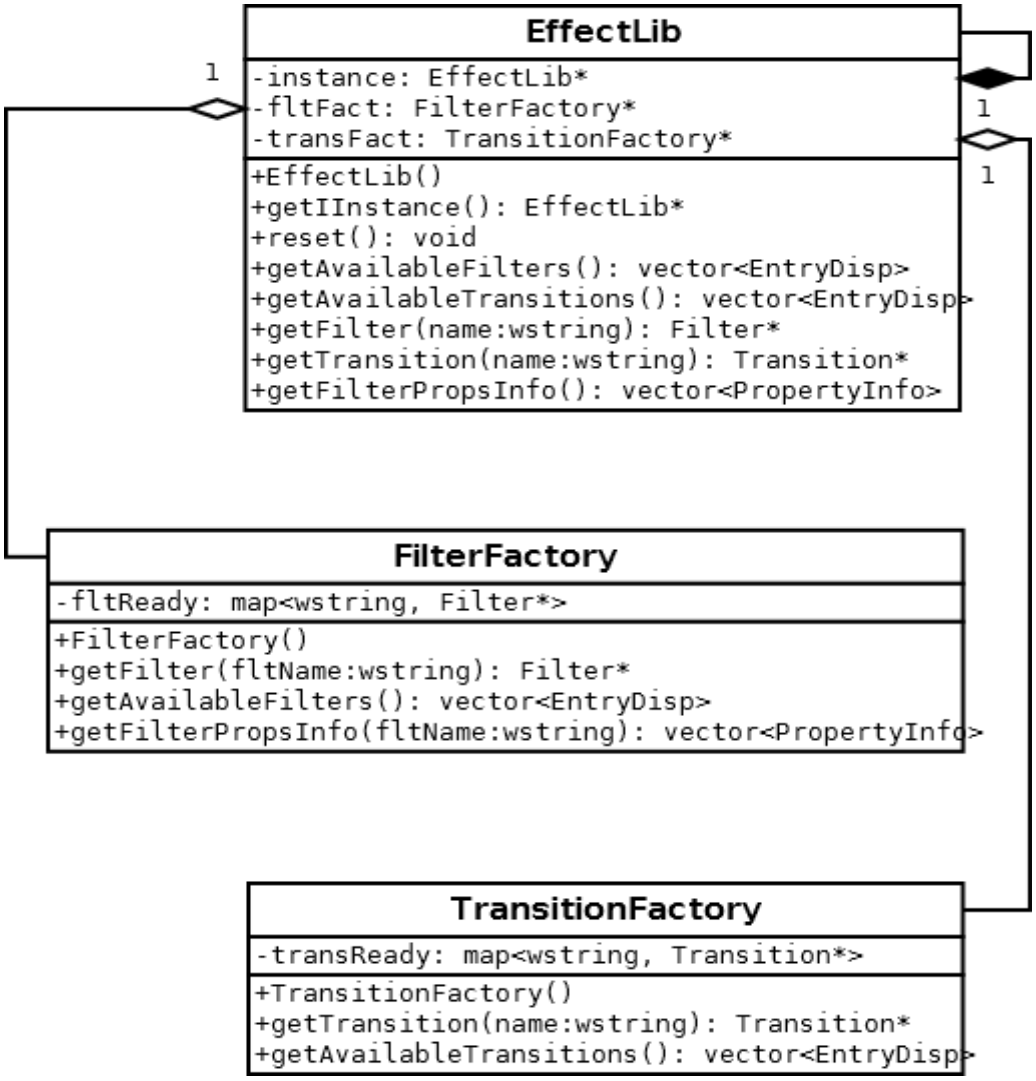

Рисунок 2.3 – Диаграмма классов для паттерна «Фасад» (класс EffectLib)

Фабрика

Порождающий паттерн «Фабрика» используется для создания объекта, принадлежащего одному из классов, на основе имеющихся некоторых данных.

Здесь используется объектами классов FilterFactory и TransitionFactory для создания объектов фильтров (объекты класса Filter и наследующих его классов) и переходов (объекты класса Transition и наследующих его) соответственно, как показано на рисунке 2.4. Для создания объекта, принадлежащего к классу какого-либо фильтра или перехода при обращении к упомянутым фабрикам достаточно лишь наименования того или иного фильтра или перехода, при условии, что то или иное

наименование находится в числе допустимых.

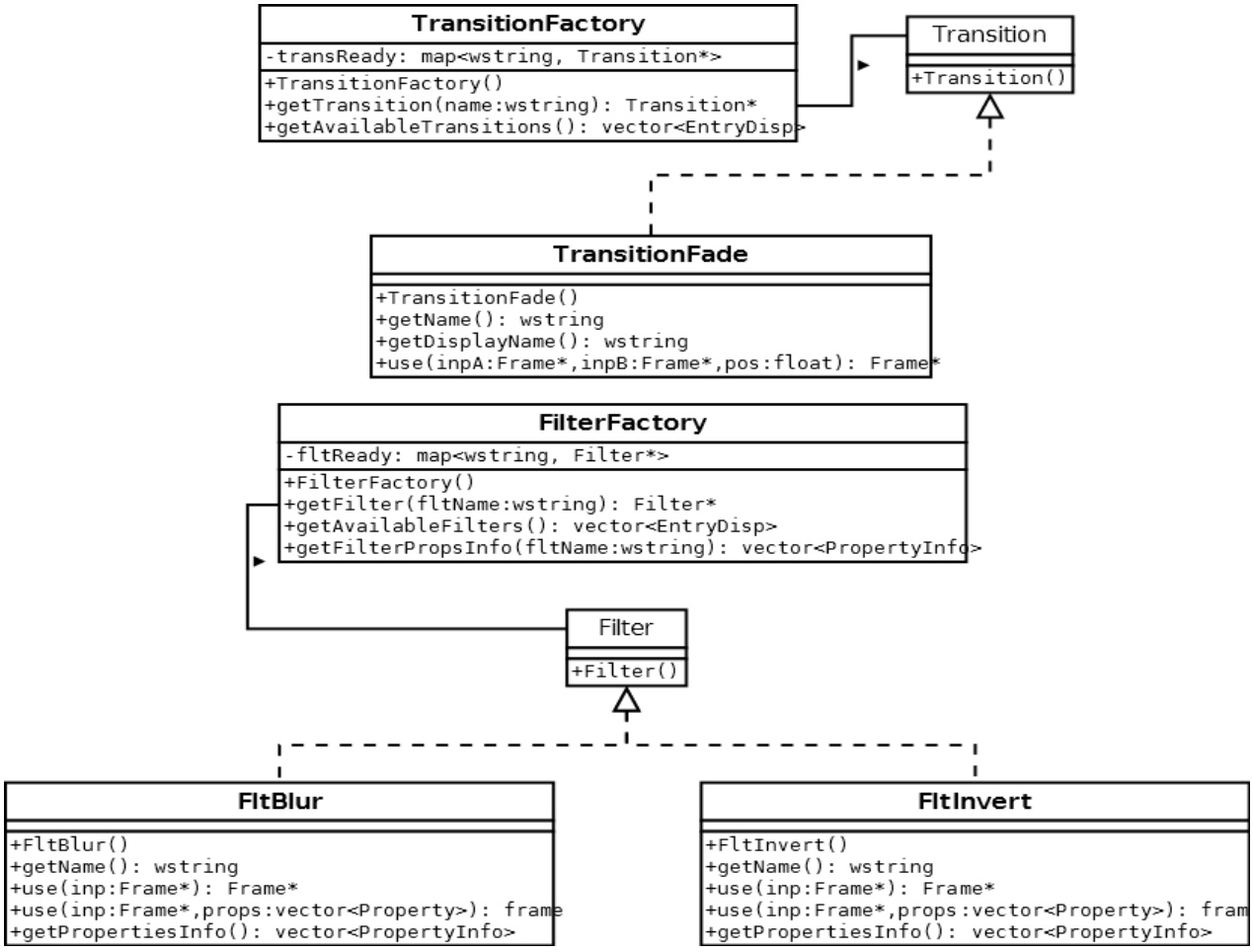

Рисунок 2.4 – Диаграмма классов для паттерна «Фабрика»

Приспособленец

Паттерн «Приспособленец» позволяет сократить количество используемых объектов разных типов, основываясь на параметризации вызовов к объектам вместо создания отдельных объектов с разными свойствами, что позволяет избежать создания новых объектов для каждого случая и тем самым помогает снизить расход используемой приложением памяти. Также допустимо определение «фабрики приспособленцев» – объекта, создающего и хранящего объекты требуемых типов и выдающего их по мере необходимости. Используется в классах фильтров (наследующихся от Filter) и переходов (наследующихся от Transition), которые вызываются при необходимости их использования, причем при их использовании предусмотрена передача параметров, определяющих их поведение при обработке

исходных данных (см. рисунок 2.4), однако в качестве обязательных параметров здесь представлены исходные и результирующие кадры.

Композит

Паттерн «Композит» используется для объединения объектов, классы которых реализуют некоторый интерфейс, в группы для одновременной обработки, содержащиеся в объекте, реализующем тот же интерфейс.

Здесь используется в объектах класса ClipCombo – клип, состоящий из двух соседних клипов, между которыми определен переход (см. рисунок 2.5). При вызове метода отрисовки, в зависимости от запрашиваемой позиции, производится выбор того, какой кадр отрисовывать – один из исходных, или кадр с примененным эффектом перехода. Такой комбинированный клип также может быть составной частью комбинированного клипа, а также к нему могут применяться видеоэффекты, как и к одиночному клипу (объекту класса ClipSingle).

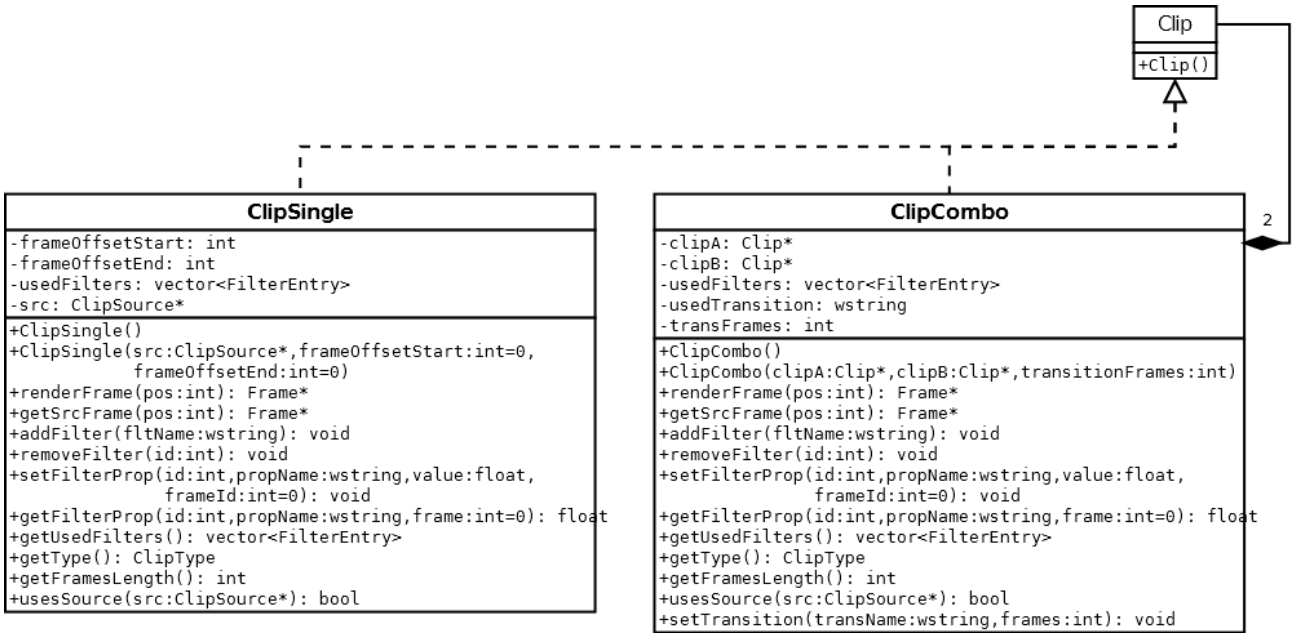

Рисунок 2.5 – Диаграмма классов для паттерна «Композит»

2.3 Диаграмма классов

На рисунке 2.6 показана общая диаграмма классов реализованного приложения.

Следует обратить внимание на то, что в данной диаграмме не отражены классы, затрагивающие интерфейс пользователя.

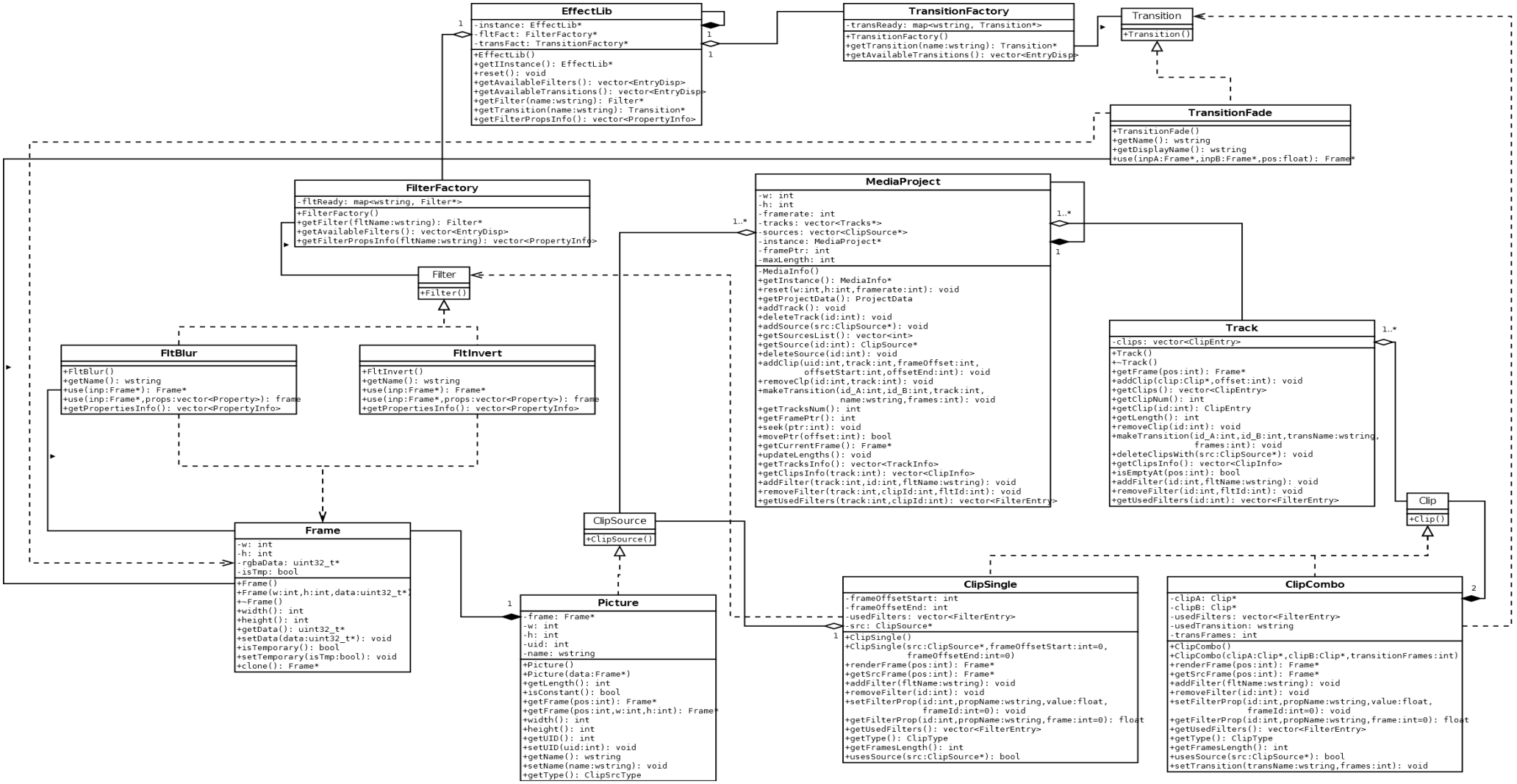

Рисунок 2.6 – Диаграмма классов

2.4 Дальнейшее развитие системы

Поскольку данная реализация является минимальной, немаловажным решением для реального использования можно считать дальнейшее расширение реализованной системы. К примеру, одним из таких расширений может стать переработка интерфейса управления имеющимися данными для более стабильной и удобной работы.

В качестве других немаловажных вариантов расширения можно рассмотреть следующие возможности:

- Поддержка фильтров и переходов с динамически изменяемыми с течением времени характеристиками, основанными, к примеру, на ключевых кадрах;
- Использование источников данных дополнительных форматов, к примеру, видеороликов, кодированных в различных форматах;
- Возможность расширения библиотеки фильтров и переходов за счет загрузки дополнительных библиотек

Также, естественно, следует уделить внимание пополнению набора доступных фильтров и переходов для большего разнообразия.

3 Результаты вычислительного эксперимента

На нижеследующих рисунках показаны примеры работы с реализованным приложением.

На рисунке 3.1 показан интерфейс реализованного приложения и пример работы с ним. В данном примере были использованы два исходных изображения, которые можно увидеть в области «Корзина». Эти изображения были добавлены на одну дорожку друг за другом. На упомянутом рисунке можно увидеть второе изображение, к которому не были применены эффекты.

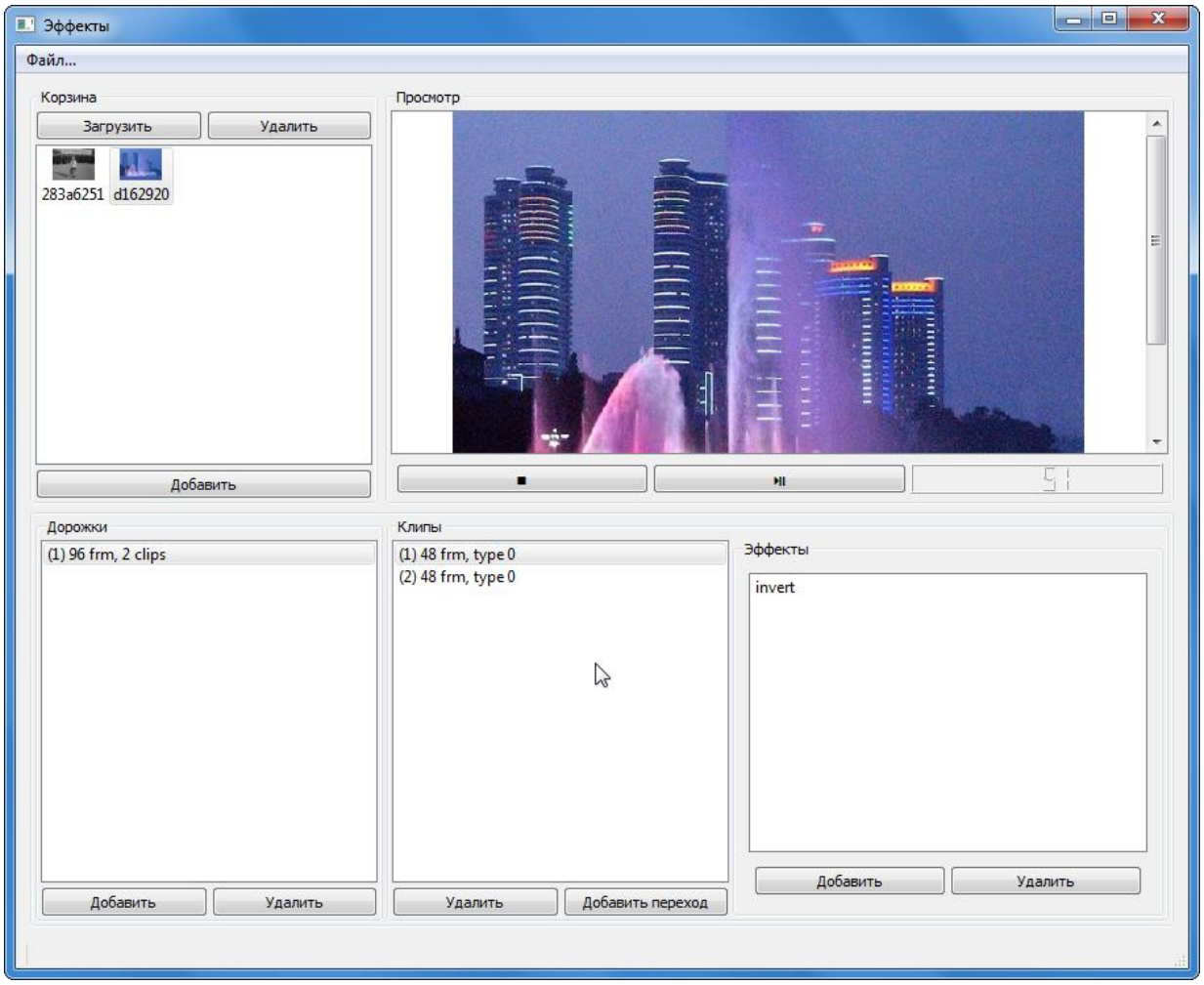

Рисунок 3.1 – Главный интерфейс приложения (два клипа на одной дорожке)

На рисунке 3.2 в качестве примера показан тот же проект, но здесь уже можно видеть первое изображение на той же дорожке, однако к нему был применен эффект инверсии.

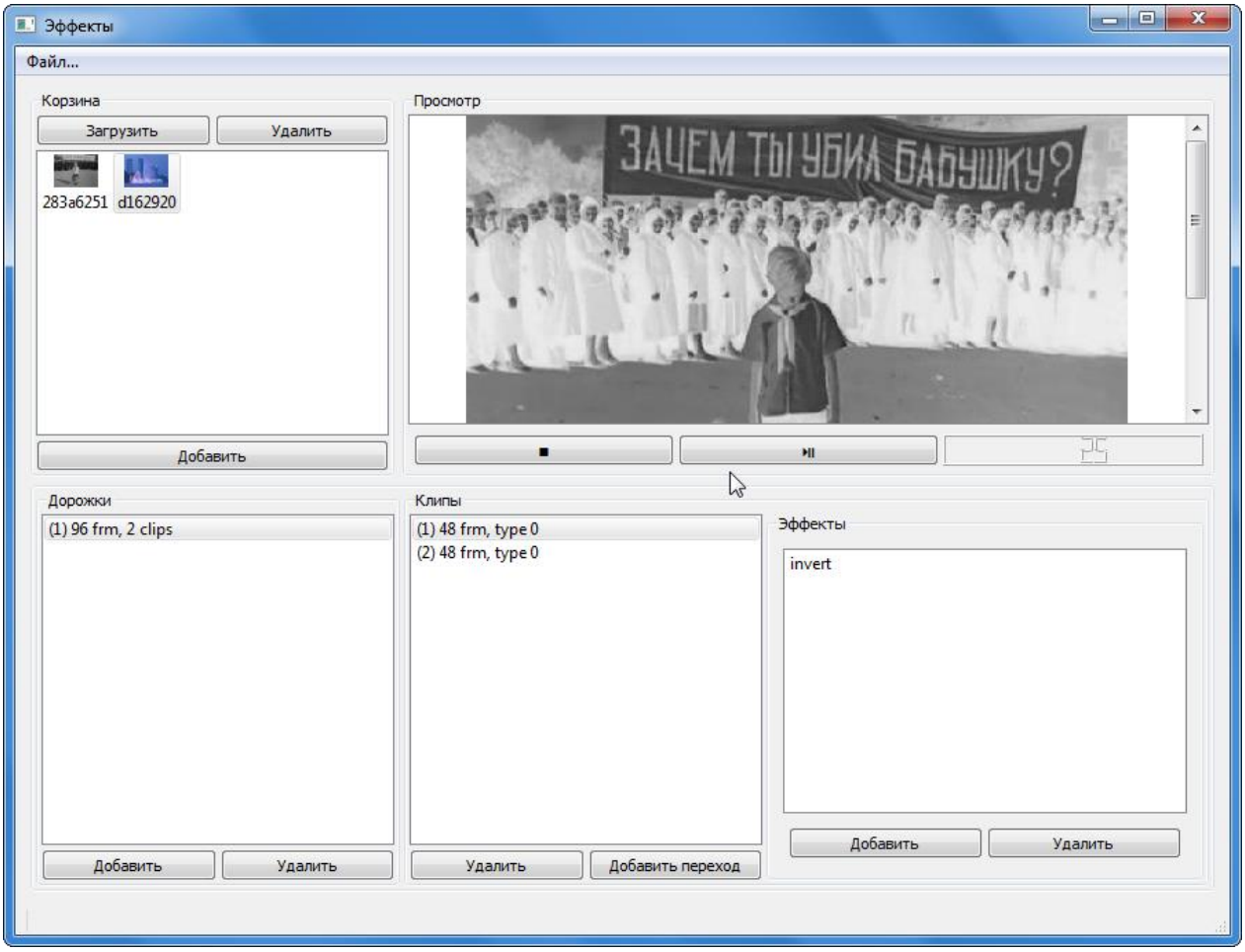

Рисунок 3.2 – Главный интерфейс приложения (тот же проект, клип с эффектом)

На рисунке 3.3 показан диалог, запрашивающий параметры для создания нового медиапроекта, а именно размер кадра в пикселях и частоту кадров.

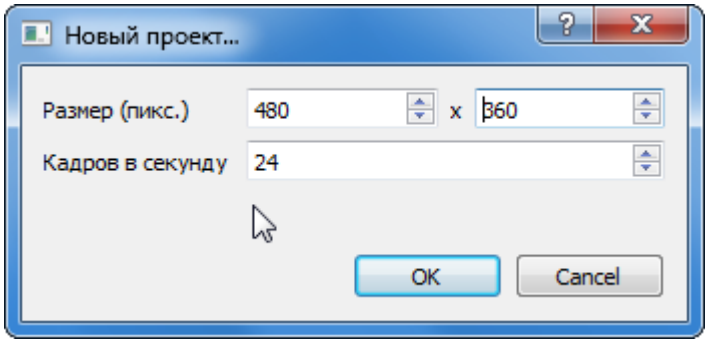

Рисунок 3.3 – Диалог создания проекта

На рисунке 3.4 показан диалог, появляющийся при добавлении того или иного исходного изображения на выбранную дорожку. Здесь предлагается задать сдвиг относительно начала дорожки или последнего клипа, находящегося на дорожке, а

также сдвиги относительно начала и конца добавляемого клипа, позволяющие его расширить или обрезать с нужной стороны. Все обозначенные здесь величины измеряются в количестве кадров.

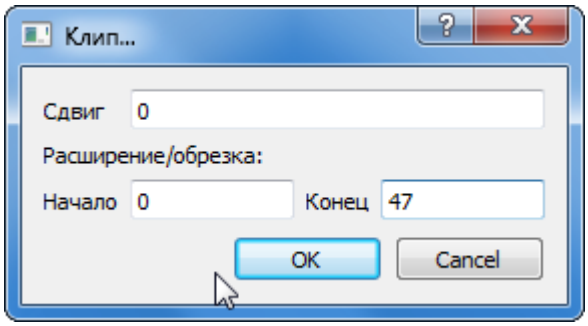

Рисунок 3.4 – Диалог добавления клипа на дорожку

На рисунке 3.5 показан диалог добавления эффекта перехода к двум стоящим рядом клипам. Здесь предлагается выбрать вид перехода и его длительность в кадрах.

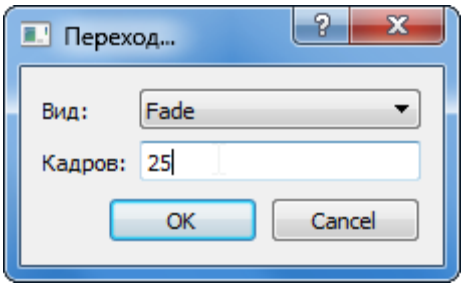

Рисунок 3.5 – Диалог добавления перехода

На рисунке 3.6 показан главный интерфейс программы после того, как к двум соседним клипам был применен эффект перехода. Следует обратить внимание, что два клипа были объединены в один комбинированный, а к нему в свою очередь был применен еще один видеоэффект.

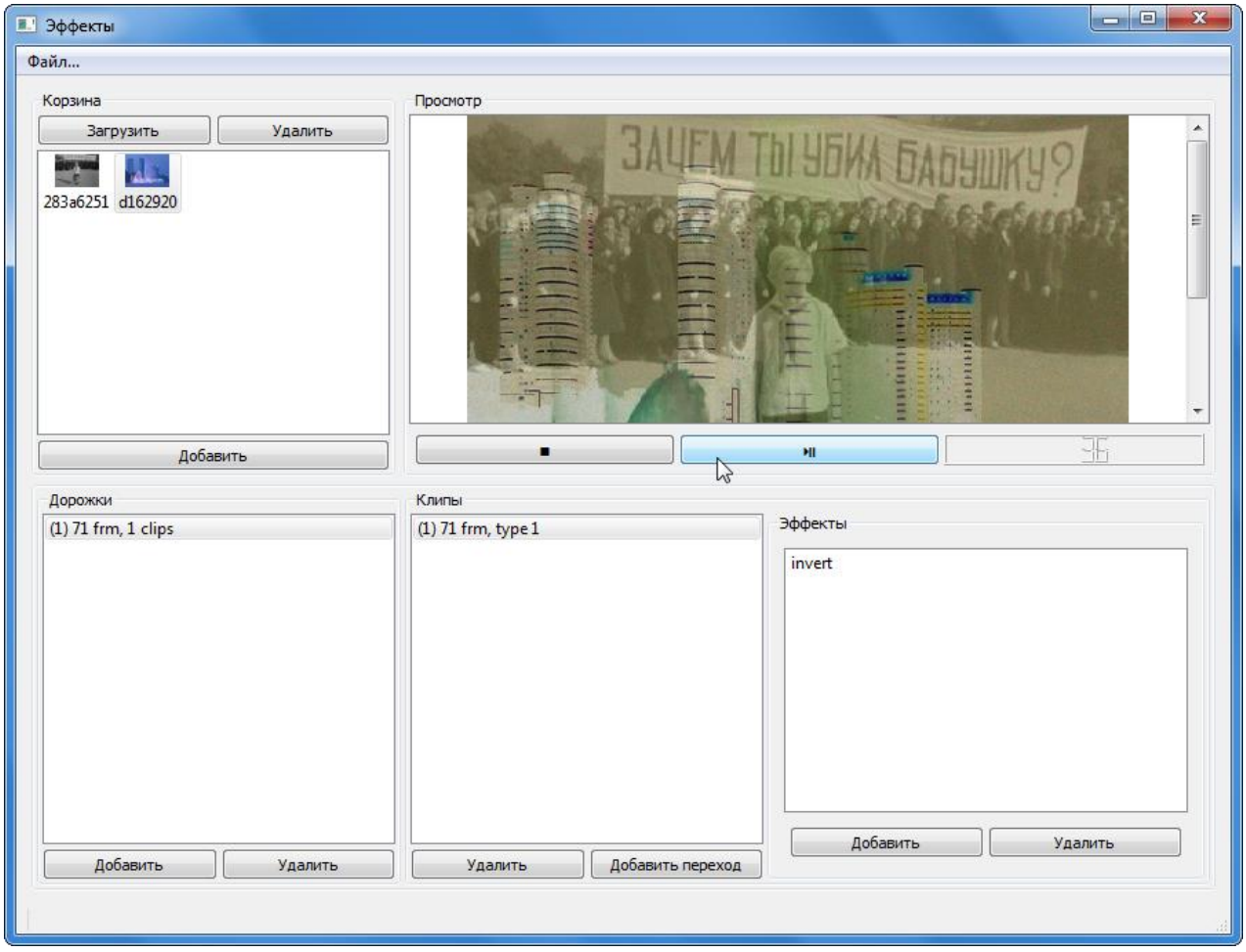

Рисунок 3.6 – Главный интерфейс программы (комбинация клипов с переходом и эффектом)

Заключение

Таким образом, в результате данной работы был спроектирован пример расширяемой библиотеки видеоэффектов и переходов, а также разработана минимальная реализация данной библиотеки и приложения, использующего ее в своей работе.

Благодаря своей структуре, в дальнейшем возможны дополнительные расширения функционала, к примеру, дополнительные виды фильтров и переходов, а также особенности их работы в зависимости от разнообразных условий.

Список использованных источников

1. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж. Приемы объектно-ориентированного проектирования. Паттерны проектирования. – СПб: Питер, 2015.
2. MLT Framework [Электронный ресурс] / – режим доступа https://www.mltframework.org, свободный.
3. Frei0r - free video effect plugin collection [Электронный ресурс] / – режим доступа https://frei0r.dyne.org, свободный.
4. The OpenFX Project [Электронный ресурс] / – режим доступа http://openfx.sourceforge.net, свободный.
5. Виктор Кудлак. Домашний видеофильм. От съемки до DVD или как расширить круг зрителей [Электронный ресурс] / – http://kudlak.ru, свободный

Приложение А. Код программы

Код программы находится на приложенном диске.