УДК 004

АРХИТЕКТУРА ПРОГРАММНОГО КОМПЛЕКСА РЕДАКТОРА ВИДЕОШАБЛОНОВ

В.А. ПЛЯСОВ
(Представлено: канд. техн. наук, доц. Д.О. ГЛУХОВ)

Рассматриваются основные моменты в архитектуре данного приложения: из каких основных частей будет построен проект, как и где будет храниться информация о видеошаблоне. Показаны система слоев в шаблоне, различные системы редактирования динамических объектов; рендеринг основного видеопотока.

Редактор видеошаблонов будет представлять приложение из нескольких модулей, т.е. клиентская часть приложения (то, что видит пользователь в браузере) серверная часть (основная архитектура приложения, база данных всех видеошаблонов) и сам редактор видеошаблонов.

На рисунке 1 представлена вся архитектура приложения.

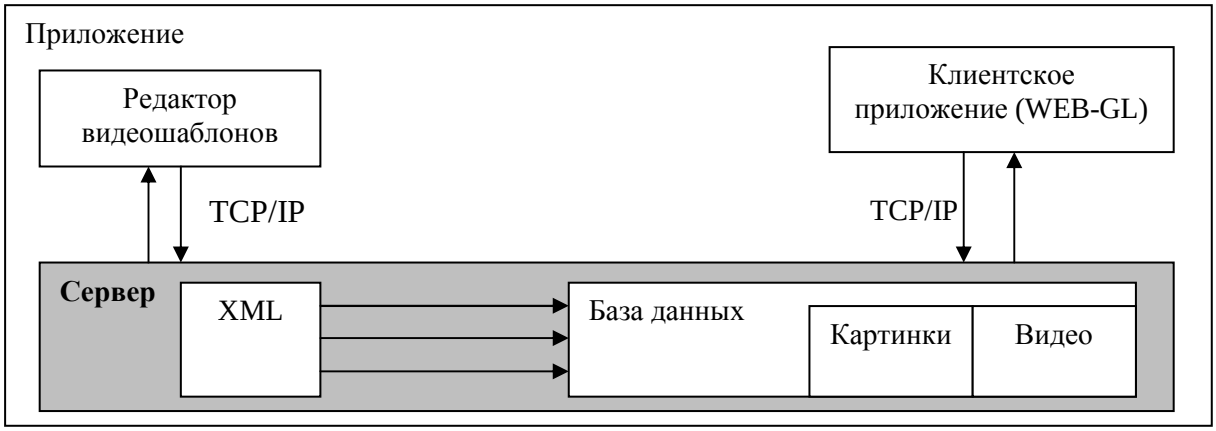

Рисунок 1. – Архитектура приложения

Сервер представляется в виде базы данных всех файлов (картинки и видео) и XML файла, который хранит в себе ссылки на данные объекты, а также список доступных видеошаблонов. Клиентское приложение представляет собой обычное WEB-приложение, которое будет через сервер работать с самой программой редактора видеошаблонов.

Основная идея заключается в том, чтобы в видео добавить различные движущиеся динамические объекты (анимация, видео и т.д.), создавая тем самым видео шаблон, т.е. есть какое-либо видео, которое загружено на сервер и есть отдельно картинки, которые необходимо добавить на данное видео. После добавления на него различных объектов, которые будут двигаться, искажаться, вращаться по мере просмотра видео, все эти данные будут сохраняться в отдельный файл типа XML – расширяемый язык разметки.

Отсюда возникла проблема, где и как на начальном этапе хранить эти данные. Чтобы упростить систему в приложении, т.е. не сразу всё загружать в XML файл, а только конечный результат, было принято решение создать отдельный класс, который будет заниматься хранением промежуточных данных, чтобы в дальнейшем всю собранную и отформатированную информацию интерпретировать в XML и тем самым создать файл видеошаблона.

Для создания более сложного шаблона необходима поддержка системы слоев, чтобы под каждый слой создавался отдельный экземпляр данного класса, который пользователь может настраивать под себя с помощью клиентского приложения.

Также возникла проблема – по какому правилу лучше всего задавать движение объектов и как их искажать между кадрами в видео. Чтобы облегчить решение данной проблемы, необходимо сдвигать и искажать не весь слой, что довольно сложно в реализации, а сдвигать вершины слоя по мере просмотра видео, т.е. сразу решается проблема с искажением (сдвинув несколько вершин слоя, – достигается искажение).

Задание ключевых кадров на видео, на которых пользователь может изменять траекторию движения объекта, искажение, вращение. На промежуточных кадрах пользователь может просмотреть, как объект выполняет все действия, если ему хочется немного изменить на данном кадре объект, то данный кадр определяется как ключевой и появляется доступ к редактированию.

Класс слоя будет хранить в себе следующую информацию:
- ссылку на объект (URL или из базы данных);
- массив из ключевых кадров, которых есть объект;
- массив из координат всех вершин объекта на ключевых кадрах;
- угол вращения объекта на ключевых кадрах;
- прозрачность объекта.

То есть в конечном итоге получаем массив объектов на видео, ссылка на видео и картинки. Затем данная информация интерполируется в XML файл (конечный видеошаблон) и загружается на сервер.

Что касается клиентского приложения, то здесь задача сводится к чтению XML-файла с видеошаблоном, создание плеера, который будет воспроизводить видео и отрисовывать все объекты на видео и функционал по редактированию объектов. Рендеринг основного видеопотока будет осуществляться параллельно с воспроизведением видео, в настоящий момент в WEB индустрии есть мощные плееры и движки, которые дают возможность осуществлять рендеринг «налету» без потерь производительности, т.е. поддерживать частоту смены кадров 30 и выше.

На рисунке 2 представлен простейший плейер с частично реалированным функционалом видеошаблонов.

Рисунок 2. – Редактор видеошаблонов

Как только видеошаблон будет создан или был загружен с сервера, пользователь может сохранить его в новое видео, в котором уже будет присутствовать данный шаблон.

В заключение проведенного исследования можно сделать **вывод**, что архитектура данного приложения является довольно сложной для реализации «в лоб».

Проанализировав всю задачу, найдены легкие способы реализации той или иной части модуля программы. Основная сложность при реализации – взаимодействие базы данных с XML файлом, который хранит список файлов и видеошаблонов, а затем выгрузка на клиентскую часть приложения.

ЛИТЕРАТУРА

1. XML формат [Электронный ресурс]. – 2016. – Режим доступа: http://okitgo.ru/xml/xml-format.html. – Дата доступа: 20.09.2016.