Реализация работы приложений, основанных на технологии виброизображения на различных платформах

С. С. Диденко, В. А. Акимов
ООО «Многопрофильное предприятие «Элсис», Санкт-Петербург, Россия,
sergey@elsys.ru

Аннотация: *в статье рассматриваются возможности по разработке и использованию программных продуктов, основанных на применении технологии виброизображения, для различных десктопных решений и структур обмена данных или мобильных устройств с различными операционными системами Android, iOS, Windows и Linux.*

Ключевые слова: *операционные системы, технология виброизображения, Android, iOS, Windows, Linux.*

Implementation of Applications Based on Vibraimage Technology on Various Platforms

Sergey S. Didenko, Valery A. Akimov
Elsys Corp, St. Petersburg, Russia,
sergey@elsys.ru

Abstract: *The article discusses the possibilities for the development and use of software products based on vibraimage applications for desktop solutions and data exchange structures or mobile devices with various operating systems Android, iOS, Windows and Linux.*

Keywords: *OS, vibraimage, Android, iOS, Windows, Linux.*

Введение

Разработка программного обеспечения, основанного на технологии виброизображения (Минкин, 2007; 2020), исторически начиналось с реализации проектов под ОС DOS и Windows. Именно для этой операционной системы были созданы флагманские продукты серии ПРО VibraImage 3, 4, 6, 7 а затем и VibraImage 10. Также для ОС Windows были созданы и более узконаправленные специализированные программы — VibraMed, VibraMid и другие.

С развитием технологии, стало очевидно, что её использование необходимо расширять на другие платформы. Этому также способствовало быстрое развитие аппаратного и программного обеспечения мобильных устройств, которые стали обладать достаточными ресурсами, чтобы обеспечивать необходимую скорость обработки информации. Как результат — в настоящее время стало возможным создавать программные продукты под наиболее популярные операционные системы

https://doi.org/10.25696/ELSYS.VC4.RU.18

для устройств: Windows, Linux, Android, iOS (рис. 1), что позволит существенно расширить количество пользователей технологией.

Рис. 1. *Операционные системы, поддерживаемые технологией виброизображения*

Приложения для ОС Windows, а также приложения и SDK для Andriod успешно распространяются уже в течение нескольких лет. Остановимся же на новых комплектах разработчика для iOS и Linux.

Создание приложений под iOS

Специальное SDK (VibraMED iOS SDK, 2021) позволяет, используя связку из самых популярных языков программирования для iOS Swift + Objective C, создавать собственное приложение с произвольным интерфейсом, используя функционал, предоставленный технологией виброизображения. В комплект поставки входит специальная библиотека, документация к ней, а также пример использования.

Съемка видео и его обработка осуществляется самим устройством. После этого данные в зашифрованном виде передаются на специальный сервер, который эти данные распаковывает и предоставляет результат пользователю (рис. 2). При этом возможны различные варианты монетизации приложения (рис. 3): по подписке, за каждый результат отдельно, платное приложение.

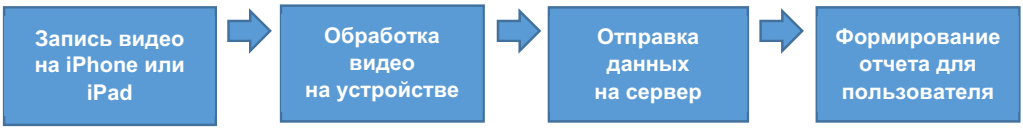

Рис. 2. *Схема работы приложения для iOS*

```
- (float)  EngineAddImage:(Byte*)data and_size:(int)size and_w:(int)w  and_h:
(int)h  and_stride:(int)stride  and_deviceRotation:(int)deviceRotation;
        Add image for processing.
        Parameters:
                data            - 8bit grayscale array of pixels
                size            - size of 'data' should be [stride*h]
                w               - image width
                h               - image height
                stride          - size of single line
                deviceRotation  - device orientation flag
        Return value: returns FPS
```

Рис. 3. *Пример описания одной из функций в SDK для iOS*

Создание приложений под Linux

Для операционной системы Linux и процессоров с архитектурой ×64 доступно SDK (рис. 4) (Vibraimage Linux SDK, 2021), включающее в себя всё необходимое для создание собственного программного продукта: бинарные и конфигурационные файлы, а также код пользовательского интерфейса.

SDK base classes and interfaces

File: VibraimageApp\CVIEngine.hpp

```
// CVIEngine is the main class for working with the SDK engine

class CVIEngine {
protected:
   CVIEngineBase * pApp;
   int          nSkip = 0;
   CVIEngine();
   bool CheckID(int id);
public:
   ~CVIEngine();
   static CVIEngine *create();
   void release();
public:
         enum VT { VT_NULL = 0, VT_INT = 1, VT_FLOAT = 2, VT_STR = 3 };

         ////////////////////////////////////////////////////////////
         // Start
         // Prform initialization and start engine
         // Parameters:
         // nCores - set thread count
         // Returns: 0
         ////////////////////////////////////////////////////////////
         int start(int nCores);

         ////////////////////////////////////////////////////////////
         // Stop:
         // Stop engine anl clear resources.
         // Returns: 0
         ////////////////////////////////////////////////////////////

   float EngineAddImage(unsigned char *data,int size,int w,int h,int stride,int deviceRotation);
   float EngineAddImage32(int *data,int size,int w,int h,int stride,int deviceRotation);
         float EngineAddImage24(unsigned char* data, int size, int w, int h, int stride, int deviceRotation);

         ////////////////////////////////////////////////////////////
         // EngineSetFace:
         // set face location
         ////////////////////////////////////////////////////////////
         int EngineSetFace( int x,int y,int w,int h);

         ////////////////////////////////////////////////////////////
         // EngineGetType:
         // Returns parameter 'type' with specified 'id' calculated with face 'nFace'
         // one of  VT_NULL, VT_INT, VT_FLOAT, VT_STR
         ////////////////////////////////////////////////////////////
   int EngineGetType(int id,int nFace = 0);

         ////////////////////////////////////////////////////////////
         // EngineGetVer:
         // Returns parameter versoin counter with specified 'id' calculated with face 'nFace'
         ////////////////////////////////////////////////////////////
         int EngineGetVer(int id, int nFace = 0);
   int EngineGetVert(const char *tag, int nFace = 0);

         ////////////////////////////////////////////////////////////
         // EngineGet*:
         // Returns parameter with specified 'id' calculated with face 'nFace'
         ////////////////////////////////////////////////////////////
   int EngineGetI(int id, int nFace = 0);
   float EngineGetF(int id, int nFace = 0);
   const char* EngineGetStr(int id);
```

Рис. 4. *Пример описания одной из функций в SDK для Linux*

При этом, логика работы приложения аналогична логике работы, описанной для iOS (рис. 3).

Новые возможности использования электронного ключа

Важной особенностью SDK (VibraMO, 2021; VibraMD, 2021) является возможность использования аппаратного ключа защиты на собственном сервере для обработки результатов (рис. 6) без обращения к серверу Elsys (рис. 5).

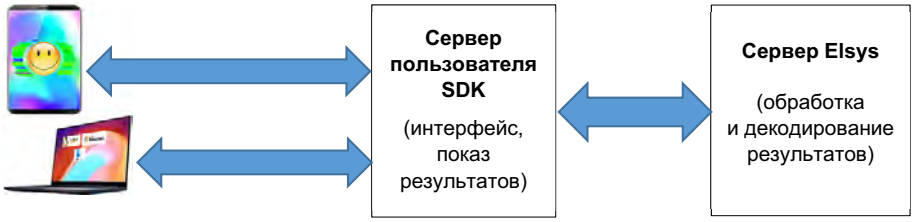

Рис. 5. *Схема работы, включающая обращение к серверу ELSYS*

Рис. 6. *Схема работы, не требующая обращений к серверу ELSYS*

Реализация подобного функционала позволяет пользователям SDK (VibraMD, 2021) создать своё приложение, работающее без обращения к серверам ELSYS для операционных систем Windows, Linux, Android, iOS. При этом для расшифровки результатов, полученных от конечных пользователей приложений, потребуется только свой собственный сервер на базе OS Windows и аппаратный ключ, поставляемый вместе с SDK (рис. 7)

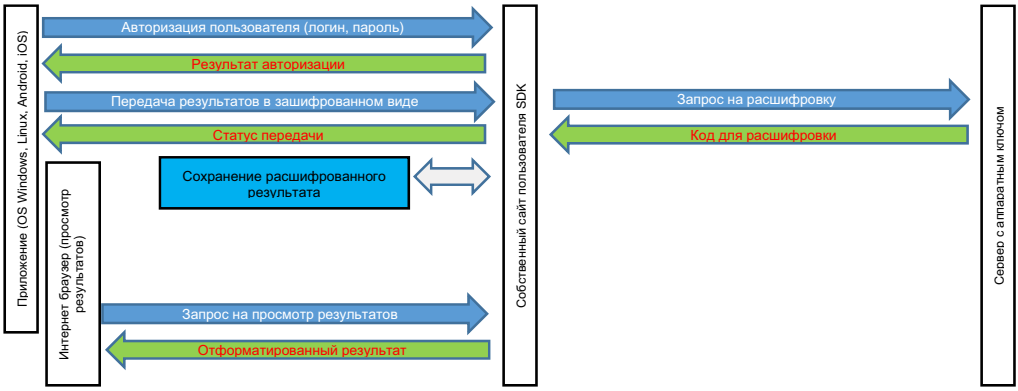

Рис. 7. *Работа приложения в составе системы с аппаратным ключом*

Заключение

Разработанные SDK для различных платформ и операционных систем позволяют существенно расширить количество пользователей технологии виброизображения за счет вовлечения владельцев устройств на ранее неподдерживаемых платформах. Кроме того, благодаря наличию комплектов разработчиков, теперь можно создать собственные приложения под наиболее популярные операционные системы.

Литература:

1. Минкин, В. А. (2007) Виброизображение. СПб.: Реноме. 108 с. https://doi.org/10.25696/ELSYS.B.RU. VI.2007
2. Минкин, В. А. (2020) Виброизображение, кибернетика и эмоции. СПб.: Реноме. 164 с. https://doi.org/10.25696/ELSYS.B.RU.VCE.2020
3. Vibraimage_Linux, (2021) Vibraimage Linux SDK description Platform [Electronic resource]. Available at: https://psymaker.com/downloads/ Linux_MIX_SDK_210316.zip (Access: 02 April 2021).
4. VibraMED_iOS, (2021) VibraMED iOS SDK Description [Electronic resource] Available at: https://psymaker.com/downloads/AppleVibraMO_SDK_20210329.zip (Access: 02 April 2021).
5. VibraMD, (2021) VibraMD SDK Description [Electronic resource] Available at: https://psymaker.com/downloads/AndroidVibraMD_SDK_210330.zip (Access: 12 April 2021).
6. VibraMO, (2021) VibraMO SDK Description [Electronic resource] Available at: https://psymaker.com/downloads/AppleVibraMO_SDK_20210329.zip (Access: 13 April 2021).