Руководство программиста

AutoSDK

Версия 2.5.x

vit

зарождая технологии™

ОГЛАВЛЕНИЕ

1 термины 4

2 обзор autosdk 5
- 2.1 Общая информация . . . 5
- 2.2 Системные требования . . . 6
- 2.3 Режимы работы продукта . . . 6
  - 2.3.1 AutoSDK Lite (“Parking”) . . . 6
  - 2.3.2 AutoSDK (“Freeflow”) . . . 6
- 2.4 Лицензирование продукта . . . 6
- 2.5 Состав продукта . . . 7
- 2.6 Процедура установки (для ОС Windows) . . . 9

3 первое использование 10
- 3.1 Загрузка модулей . . . 10
- 3.2 Создание принципала . . . 10
- 3.3 Конфигурация принципала . . . 11
- 3.4 Распознавание . . . 12
- 3.5 Обработка результата распознавания . . . 13

4 описание вспомогательных утилит 16
- 4.1 Утилита vpwfetch . . . 16
  - 4.1.1 Обзор . . . 16
  - 4.1.2 Семантика выражений . . . 16
  - 4.1.3 Настройка . . . 20
  - 4.1.4 Завершение работы и примеры команд . . . 24
- 4.2 Утилита lsvpwi . . . 24
  - 4.2.1 Обзор . . . 24
  - 4.2.2 Опции и примеры . . . 25
- 4.3 Утилита lsvpwc . . . 27
  - 4.3.1 Обзор . . . 27
  - 4.3.2 Опции . . . 27

5 описание операций с aorp-объектами 29
- 5.1 VpwprincOpen . . . 29
  - 5.1.1 Имя . . . 29
  - 5.1.2 Синтаксис . . . 29
  - 5.1.3 Описание . . . 29
  - 5.1.4 Возвращаемые значения . . . 30
- 5.2 VpwprincOpen_v3 . . . 31
  - 5.2.1 Имя . . . 31
  - 5.2.2 Синтаксис . . . 31
  - 5.2.3 Описание . . . 31

5.2.4 Возвращаемые значения . . . 32
5.3 AorpRetain . . . 33
5.3.1 Имя . . . 33
5.3.2 Синтаксис . . . 33
5.3.3 Описание . . . 33
5.3.4 Возвращаемые значения . . . 33
5.4 AorpRelease . . . 34
5.4.1 Имя . . . 34
5.4.2 Синтаксис . . . 34
5.4.3 Описание . . . 34
5.4.4 Возвращаемые значения . . . 34
5.5 VodiprincGetparam . . . 35
5.5.1 Имя . . . 35
5.5.2 Синтаксис . . . 35
5.5.3 Описание . . . 35
5.5.4 Возвращаемые значения . . . 35
5.6 VodiprincSetparam . . . 36
5.6.1 Имя . . . 36
5.6.2 Синтаксис . . . 36
5.6.3 Описание . . . 36
5.6.4 Возвращаемые значения . . . 41
5.7 VodiprincApplyparam . . . 42
5.7.1 Имя . . . 42
5.7.2 Синтаксис . . . 42
5.7.3 Описание . . . 42
5.7.4 Возвращаемые значения . . . 42
5.8 VodiprincControl . . . 43
5.8.1 Имя . . . 43
5.8.2 Синтаксис . . . 43
5.8.3 Описание . . . 43
5.8.4 Возвращаемые значения . . . 44
5.9 VodiprincProcess . . . 45
5.9.1 Имя . . . 45
5.9.2 Синтаксис . . . 45
5.9.3 Описание . . . 45
5.9.4 Возвращаемые значения . . . 46
5.10 VodiprincFlush . . . 47
5.10.1 Имя . . . 47
5.10.2 Синтаксис . . . 47
5.10.3 Описание . . . 47
5.10.4 Возвращаемые значения . . . 47
5.11 VodiensCount . . . 48
5.11.1 Имя . . . 48
5.11.2 Синтаксис . . . 48
5.11.3 Возвращаемые значения . . . 48
5.12 VodiensClear . . . 49

5.12.1 Имя . . . 49
5.12.2 Синтаксис . . . 49
5.12.3 Описание . . . 49
5.12.4 Возвращаемые значения . . . 49
5.13 VodiensAdd . . . 50
5.13.1 Имя . . . 50
5.13.2 Синтаксис . . . 50
5.13.3 Описание . . . 50
5.13.4 Возвращаемые значения . . . 50
5.14 VodiensRemove . . . 51
5.14.1 Имя . . . 51
5.14.2 Синтаксис . . . 51
5.14.3 Описание . . . 51
5.14.4 Возвращаемые значения . . . 51
5.15 VodiensDelete . . . 52
5.15.1 Имя . . . 52
5.15.2 Синтаксис . . . 52
5.15.3 Описание . . . 52
5.15.4 Возвращаемые значения . . . 52
5.16 VodiensGet . . . 53
5.16.1 Имя . . . 53
5.16.2 Синтаксис . . . 53
5.16.3 Описание . . . 53
5.16.4 Возвращаемые значения . . . 53
5.17 VodiensUnique . . . 54
5.17.1 Имя . . . 54
5.17.2 Синтаксис . . . 54
5.17.3 Описание . . . 54
5.17.4 Возвращаемые значения . . . 54
5.18 VodiensCombine . . . 55
5.18.1 Имя . . . 55
5.18.2 Синтаксис . . . 55
5.18.3 Описание . . . 55
5.18.4 Возвращаемые значения . . . 55
5.19 VodiensCombine_v2 . . . 56
5.19.1 Имя . . . 56
5.19.2 Синтаксис . . . 56
5.19.3 Описание . . . 56
5.19.4 Возвращаемые значения . . . 56
5.20 VodiresGetuserdata . . . 57
5.20.1 Имя . . . 57
5.20.2 Синтаксис . . . 57
5.20.3 Описание . . . 57
5.20.4 Возвращаемые значения . . . 57
5.21 VodiresSetuserdata . . . 58
5.21.1 Имя . . . 58

5.21.2 Синтаксис . . . 58
5.21.3 Описание . . . 58
5.21.4 Возвращаемые значения . . . 58
5.22 VodiresFetchinfo . . . 59
5.22.1 Имя . . . 59
5.22.2 Синтаксис . . . 59
5.22.3 Описание . . . 59
5.22.4 Возвращаемые значения . . . 59
5.23 VodiResultInfoDestroy . . . 60
5.23.1 Имя . . . 60
5.23.2 Синтаксис . . . 60
5.23.3 Описание . . . 60

6 описание структур 61
6.1 vodi_vpw_country_id . . . 61
6.1.1 Имя . . . 61
6.1.2 Синтаксис . . . 61
6.1.3 Описание . . . 61
6.2 vodi_image . . . 62
6.2.1 Имя . . . 62
6.2.2 Синтаксис . . . 62
6.2.3 Описание . . . 62
6.3 vodi_vpw_options . . . 63
6.3.1 Имя . . . 63
6.3.2 Синтаксис . . . 63
6.3.3 Описание . . . 63
6.4 vodi_ucontext . . . 64
6.4.1 Имя . . . 64
6.4.2 Синтаксис . . . 64
6.4.3 Описание . . . 64
6.5 vodi_plate_info_spec . . . 65
6.5.1 Имя . . . 65
6.5.2 Синтаксис . . . 65
6.5.3 Описание . . . 66
6.6 vodi_plate . . . 68
6.6.1 Имя . . . 68
6.6.2 Синтаксис . . . 68
6.6.3 Описание . . . 68
6.7 aorp_error . . . 70
6.7.1 Имя . . . 70
6.7.2 Синтаксис . . . 70
6.7.3 Описание . . . 70

7 описание функций для работы с шаблонами 71
7.1 VodiISO3166children . . . 71
7.1.1 Имя . . . 71
7.1.2 Синтаксис . . . 71

7.1.3 Описание . . . 71
7.1.4 Возвращаемые значения . . . 71
7.2 LpvlibShowPlateid . . . 72
7.2.1 Имя . . . 72
7.2.2 Синтаксис . . . 72
7.2.3 Описание . . . 72
7.2.4 Возвращаемые значения . . . 72
7.3 LpvlibReadPlateid . . . 73
7.3.1 Имя . . . 73
7.3.2 Синтаксис . . . 73
7.3.3 Описание . . . 73
7.3.4 Возвращаемые значения . . . 73
7.4 LpvlibModulesOf . . . 74
7.4.1 Имя . . . 74
7.4.2 Синтаксис . . . 74
7.4.3 Описание . . . 74
7.4.4 Возвращаемые значения . . . 75

8 описание грамматики для обозначения множества шаблонов номерных пластин 76

1 ТЕРМИНЫ

- Принципал (**Vpwprinc**) — объект, для которого определены операции распознавания автомобильных номеров (**VodiprincProcess**). Одному принципалу соответствует один канал распознавания. Распознавание по данному каналу может быть разделено на несколько параллельных нитей (далее — потоков распознавания, англ. “recognition threads”), чтобы оптимально распределить нагрузку на процессор при обработке видео высокого разрешения.
- Результат (**Vpwres**) — объект, описывающий результат распознавания автомобильного номера.
- Ансамбль (**Vpwens**) — список результатов распознавания автомобильных номеров.

  > **Vpwprinc** (принципал), **Vpwens** (ансамбль) и **Vpwres** (результат) относятся к множеству так называемых AORP-объектов.

- Общая библиотека, динамическая библиотека (**shared library**) — библиотека общего использования.
- Архивная библиотека, статическая библиотека (**archive library**) — библиотека архивного типа.
- Библиотека заготовок, стабовая библиотека (**stub library**) — библиотека, используемая в ОС Windows для линковки с динамической библиотекой.

2 ОБЗОР AUTOSDK

2.1 общая информация

Комплект разработчика **AutoSDK** (VIT) — набор динамических библиотек и заголовочных файлов C/C++. На основе **AutoSDK** разрабатывается программное обеспечение для распознавания автомобильных номеров. Данная функциональность может использоваться как компонент следующих систем:

- системы контроля доступа;
- системы управления парковкой;
- системы автоматического сбора пошлины[1];
- системы “умных городов”;
- системы видеонаблюдения транспортных потоков;
- системы управления видео[2];
- системы взвешивания (в т.ч. в движении) и контроля нагрузки на ось;
- встроенные устройства и программно-аппаратные комплексы.

Возможности **AutoSDK**:

- видеоанализ в режиме реального времени со скоростью обработки кадра в диапазоне 10-100 мс (в зависимости от разрешения кадра);
- распознавание двухстрочных и однострочных номерных знаков более 30 стран[3];
- распознавание до 20 номеров в одном кадре (для видеонаблюдения за несколькими полосами движения по одному каналу);
- максимальная скорость движения автомобиля, при которой распознается номер — 220 км/час (при корректно настроенной камере видеонаблюдения);
- определение направления движения транспортного средства;
- гибкая настройка распознавания: выбор шаблонов номеров, задание нескольких зон детекци номера в кадре, управление параметрами выдачи результата;
- выбор наилучшего результата распознавания из множества результатов, сгенерированных во время наблюдения за конкретным номеров в кадре (алгоритм трекинга).

[1] англ. “electronic toll collection system”, ETC
[2] англ. “video management system”, VMS
[3] актуальный список государств, номерные знаки которых могут распознаваться с помощью **AutoSDK**, предоставляется по запросу

2.2 системные требования

- **Поддерживаемые операционные системы:** семейство Unix (например, Mac OS X, Linux, FreeBSD) и Microsoft Windows (например, 2000, 2003, XP, Vista, Seven, 8/8.1, 10).
- **Поддерживаемые архитектуры процессора:** x86 (x86-32, x86-64), ARMv7, ARMv8 AArch32.
- **Модель процессора:** подбирается индивидуально в соответствии с планируемым количеством потоков распознавания, кадровой частотой обрабатываемого видео и разрешением кадра. При активации дополнительных потоков распознавания нагрузка на процессор растет линейно.
- **Оперативная память:** от 2 Гб. Объем памяти, потребляемой одним потоком распознавания, зависит от кадровой частоты видео и разрешения кадра, и приблизительно равняется 200 Мб.
- **Языки программирования:** C, C++.
- **Минимальная высота символов на изображении номерной пластины в кадре:**
  - 14-20 пикселей для камер без аппаратного сжатия кадра (аналоговых, машинного зрения);
  - 20-30 пикселей для камер с аппаратным сжатием кадра (IP-камер).
- **Оптимальные углы инсталляции камеры видеонаблюдения:**
  - по вертикали — 18-20 (максимум — 30) градусов;
  - по горизонтали — 5-10 (максимум — 20) градусов;
- **Допустимый крен номерной пластины в кадре:** 5-10 градусов.
- **Входные данные:** 8-битный RAW-поток (в градациях серого[4]).

2.3 режимы работы продукта

2.3.1 AutoSDK Lite (“Parking”)

- Обрабатывает до 6 кадров в секунду (англ. *fps*, frames per second).
- Требуемая скорость движения транспортных с средств в зоне видеоконтроля: до 20 км/ч

2.3.2 AutoSDK (“Freeflow”)

- Обрабатывает до 25 кадров в секунду.
- Требуемая скорость движения транспортных с средств в зоне видеоконтроля: до 220 км/ч

2.4 лицензирование продукта

Динамические библиотеки **AutoSDK** защищены и могут использоваться только при наличии локально установленных ключей защиты[5] (см. таблицу 1).

[4] англ. “grayscale”

[5] лицензионная защита **AutoSDK** осуществляется с помощью решения Sentinel HASP компании Gemalto

Таблица 1: Особенности ключа защиты **Sentinel HASP** в зависимости от его типа

Тип ключа	Описание
HASP HL Key (англ. “hardware license”, аппаратная защита)	Реализована в виде аппаратного токена (USB-ключа). Физически устанавливается на сервер с защищаемыми программными продуктами. Может переноситься без потери лицензии.
HASP SL Key (англ. “software license”, программная защита)	Реализована в виде сервиса. Привязывается к аппаратному обеспечению компьютера. В случае переноса защищаемых продуктов на другой сервер необходимо приобрести новую лицензию.

От приобретенной лицензии зависят следующие параметры работы **AutoSDK** :

- максимальное количество работающих потоков распознавания (по всем каналам);
- доступные шаблоны номеров конкретных стран;
- скорость обработки видеопотока (6 fps или до 25 fps, см. Режимы работы продукта).

> Просмотреть параметры лицензии к **AutoSDK**, которая доступна на локальном компьютере, можно с помощью утилиты lsvpwc (см. Утилита lsvpwc).

2.5 состав продукта

В состав дистрибутива **AutoSDK** входят файловый пакет и ключ защиты (разрешающий работу исполняемых файлов). Структура файлового пакета:

- **_doc**:
  - AutoSDK_v2.5_RU.pdf, AutoSDK_v2.5_EN.pdf — документация разработчика (на русском и английском языках);
  - AutoSDK_2.5.x_change_log_RU.pdf, AutoSDK_2.5.x_change_log_EN.pdf — история изменений (на русском и английском языках);
  - **templates** — изображения поддерживаемых шаблонов номерных пластин.
- **_examples** — примеры использования **AutoSDK**.
- **_redist** — вспомогательные программные продукты других производителей[6].
- **_opt**:
  - **include** — заголовочные файлы:
    - Vodi — заголовочные файлы **AutoSDK**;
    - Bo — заголовочные файлы и вспомогательные библиотеки (в т.ч. для портируемости программного кода).
    - ImageMagick — библиотеки обработки растровых изображений.

[6] например, окружение для работы лицензионноых ключей (Sentinel LDK Run-time Environment)

- **bin** [**Unix**] — утилиты, примеры использования;
- **bin** [**Windows**] — программы, динамические библиотеки, среди которых:
  - vpwfetch — утилита для чтения автомобильных номеров (см. Утилита vpwfetch);
  - lsvpwi — утилита для печати информации о шаблонах номеров (см. Утилита lsvpwi);
  - lsvpwc — утилита для печати информации про лицензию (см. Утилита lsvpwc);
  - Bo.dll — вспомогательная библиотека;
  - Bo_g.dll — отладочная версия;
  - Vodi.dll — основная библиотека;
  - Vodi_g.dll — отладочная версия.
- **lib** [**Unix**] — объектные файлы, динамические и архивные библиотеки:
  - vpwfetch — утилита для чтения автомобильных номеров (см. Утилита vpwfetch);
  - lsvpwi — утилита для печати информации о шаблонах номеров (см. Утилита lsvpwi);
  - lsvpwc — утилита для печати информации про лицензию (см. Утилита lsvpwc);
  - [PFX]Bo[SFX] — вспомогательная библиотека;
  - [PFX]Bo_g[SFX] — отладочная версия;
  - [PFX]Vodi[SFX] — основная библиотека;
  - [PFX]Vodi_g[SFX] — отладочная версия.
- **lib** [**Windows**] — объектные файлы, архивные и стабовые библиотеки, среди которых:
  - Bo.lib — стабовая библиотека Bo.dll;
  - Bo_g.lib — отладочная версия;
  - Vodi.lib — стабовая библиотека Vodi.dll;
  - Vodi_g.lib — отладочная версия.
- **libexec** — библиотеки с неявной линковкой.

В составе названия динамической библиотеки для Unix могут использоваться:

- [PFX] — системно-зависимый префикс (например, lib);
- [SFX] — системно-зависимый суффикс, включающий версию файла и его расширение (например, .so.2.5.0; 2.5.0.dylib, -2_5_0.dll).

В папке **_doc\templates** находятся папки, содержащие в своем названии конкретную страну и ее код согласно ISO 3166-1 numeric. Каждая из папок содержит изображения шаблонов номерных пластин страны, заявленной в названии. Имена изображений соответствуют порядковым номерам шаблонов в пределах модуля страны (**vpwi-[country]**)[a].

[a] где [country] — код этой страны (например, "vpwi-co" используется для распознавания автомобильных номеров Колумбии).

Соответствие старых и новых идентификаторов шаблонов номерных пластин[a] каждой страны записано в файле **doc/templates/templates_map.txt**. Альтернативой файлу является консольная утилита lsvpwi (см. Утилита lsvpwi).
Форма записи: ID_страны[.ID_региона].ID_шаблона:прежний_ID_шаблона

[a] новые идентификаторы реализованы в версии **AutoSDK** 2.5.1

2.6 процедура установки (для ос windows)

Рекомендуется придерживаться данной инструкции для корректной установки **AutoSDK**.

> Пусть INSTALLDIR — путь к директории установки продукта.

1. Запустить исполняемый файл **setup.exe** (находится в корне файлового пакета). Пройти шаги программы-инсталлятора. В результате файлы **AutoSDK** будут помещены в INSTALLDIR.
2. Добавить для препроцессора C/C++ путь поиска заголовочных файлов (INSTALLDIR\opt\include).
3. Добавить для линковщика путь поиска библиотек (INSTALLDIR\opt\lib).

   > При использовании Microsoft Visual Studio это делается в разделе Additional Include Directories.

4. Создать или добавить в переменную окружения PATH путь INSTALLDIR\opt\bin.
5. Создать или добавить в переменную окружения AORP_MODULE_PATH путь INSTALLDIR\opt\libexec\aorp\modules.

3 ПЕРВОЕ ИСПОЛЬЗОВАНИЕ

3.1 загрузка модулей

Перед началом работы с **AutoSDK** необходимо загрузить 2 модуля:

- **vpw** — базовая функциональность;
- **vpwi-[country]** — описание номерных пластин конкретной страны (например, "vpwi-co" для Колумбии).

```
static bo_status_t _t_load_requireds(struct aorp_error *anErrPtr)
{
  static char const *_s_requireds[] = { "vpw", "vpwi-co", NULL };

  bo_status_t status;
  char const *name, **rqp;

  for (rqp = _s_requireds; NULL != (name = *rqp); ++rqp) {
    status = AorpMldLoad(name, NULL, 0, 0, anErrPtr);
    if (BoS_FAILURE(status)) {
      return (status);
      /* NOTREACHED */
    }
  }

  return (BoS_NORMAL);
}
```

> См. дополнительную информацию в разделе:
>
> - aorp_error

3.2 создание принципала

Чтобы начать распознавание автомобильных номеров, необходимо создать (открыть) экземпляр принципала (см. термины). При создании указываются параметры работы принципала.

```
int main(int argc, char *argv[])
{
    aorp_object_t principal;
    struct aorp_error *err;
    ...
    principal = VpwprincOpen(0, NULL, NULL, err);
```

```
    if (NULL == principal) {
        goto L_error;
        /* NOTREACHED */
    }
    ...
    (void)AorpRelease(principal, 0, NULL); /* releasing the principal */
    ...
}
```

Чтобы среди всех доступных лицензионных ключей использовать тот, который предоставляет наибольшую скорость обработки видео, необходимо при открытии принципала указать для параметра **aFpsMax** значение 0.

См. дополнительную информацию в разделах:

- VpwprincOpen
- AorpRelease
- aorp_error

3.3 конфигурация принципала

После создания принципала можно приступить к его настройке. Из примера ниже видно, что распознаваться должны номерные знаки Колумбии.

```
static bo_status_t
_t_princ_configure(aorp_object_t aPrincipal, struct aorp_error *anErrPtr)
{
    bo_status_t status;
    struct vodi_vpw_country_id tmpl = {
        /* .code = */170/* co */,
        /* .id = */0,
        /* .sub_id = */0
    };

    assert(NULL != aPrincipal);

    status = VodiprincSetparam(aPrincipal, anErrPtr,
            VodiCTL_VPW_PLATE_TEMPLATE, &tmpl, 1);
    if (BoS_FAILURE(status)) {
        return (status);
        /* NOTREACHED */
    }

    /* other parameters */

```

```
    status = VodiprincApplyparam(result, anErrPtr);
    return (status);
}
```

См. дополнительную информацию в разделах:

- VodiprincSetparam
- VodiprincApplyparam
- vodi_vpw_country_id
- aorp_error

3.4 распознавание

Для того, чтобы начать распознавание, необходимо передать изображение (запакованное в структуру **vodi_image**) в операцию **VodiprincProcess**. Каждая строка изображения должна быть выровнена на 4 байта [1].
Структура **vodi_vpw_options** с дополнительными опциями (номер кадра в последовательности, его временная метка и размер) используется при распознавании очереди кадров.
В случае обнаружения автомобильных номеров на изображении операция вернет ансамбль результатов (см. термины).

```
static bo_status_t
_t_princ_recognize(
    aorp_object_t aPrincipal,
    struct vodi_image *anImage,
    size_t anAnalizeZonec,
    vodi_rect_t const anAnalizeZonev[],
    struct vodi_ucontext *anUserCtx,
    aorp_object_t anInputEnsemble,
    aorp_object_t *anOutputEnsemblePtr,
    struct aorp_error *anErrPtr
    )
{
    bo_status_t status;
    struct vodi_vpw_options options;

    options.magic = VodiK_VPW_OPTIONS_WORK_MAGIC;
    options.img_seqnum = 0;
    options.img_timestamp = 0;
    options.imgsz.sz_width = anImage->img_width;
    options.imgsz.sz_height = anImage->img_height;

    status = VodiprincProcess(aPrincipal, anImage, anAnalizeZonec,
```

[1] См. макрофункцию VODI_IMAGE_WStride (_opt\include\Vodi\Types.h). Макрофункция формально определяет, что шаг по строкам изображения — это ширина изображения, выровненная на 4.

```
        anAnalizeZonev, anUserCtx, anInputEnsemble, anOutputEnsemblePtr,
        &options, anErrPtr);

    return (status);
}
```

См. дополнительную информацию в разделах:

- VodiprincProcess
- vodi_vpw_options
- vodi_image
- aorp_error

3.5 обработка результата распознавания

Обработка полученного результата начинается с чтения из ансамбля.

```
static bo_status_t
_t_ensemble_handle(aorp_object_t anEnsemble, struct aorp_error *anErrPtr)
{
    bo_status_t status;
    size_t resc;
    aorp_object_t resv[256], *resp;
    bo_index_t from;
    _Bool overflow;

    assert(NULL != anEnsemble);

    from = 0;
    do {
        status = VodiensGet(
            anEnsemble, from, -1, __myc_namembs(resv), resv, anErrPtr
        );
        if (!BoS_DOBRE(status))
            break;

        resc = (size_t)status;
        overflow = false;
        if (__myc_namembs(resv) < resc) {
            resc = __myc_namembs(resv);
            from += (bo_index_t)resc;
            overflow = true;
        }

```

```
        resp = resv;
        do {
            status = _t_result_handle(*resp++, anErrPtr);
            if (BoS_FAILURE(status)) {
                return (status);
                /* NOTREACHED */
            }
        }
        while (--resc);
    }
    while (overflow);

    return (BoS_NORMAL);
}
```

К каждому результату из ансамбля нужно применить функцию **_t_result_handle**. Она использует операцию **VodiresFetchinfo**, чтобы прочитать информацию из результата в структуру **vodi_vpw_result_info**. По завершении работы с прочитанной информацией её необходимо разрушить с помощью функции **VodiResultInfoDestroy**.

```
static bo_status_t
_t_result_handle(aorp_object_t aResult, struct aorp_error *anErrPtr)
{
    bo_status_t status;
    struct vodi_result_info *info;
    struct vodi_vpw_result_info plateinfo;

    assert(NULL != aResult);

    info = (struct vodi_result_info *)&plateinfo;
    VODI_RESULT_INFO_TYPE(info) = VodiK_VPW_RESULT_INFO;
    status = VodiresFetchinfo(aResult, info, anErrPtr);
    if (BoS_FAILURE(status)) {
        return (status);
        /* NOTREACHED */
    }

    status = _t_plateinfo_handle(&plateinfo, anErrPtr);

    VodiResultInfoDestroy(info);

    return (status);
}
```

К прочитанной информации применяется функция **_t_plateinfo_handle**, с помощью которой можно произвести определенные действия над полученным результатом (например, распечатать текст номерной пластины).

```
static bo_status_t
_t_plateinfo_handle(
```

```
    struct vodi_vpw_result_info *anInfo,
    struct aorp_error *anErrPtr
    )
{
    struct vodi_plate_info_spec *pis;
    struct vodi_plate *plate;

    assert(NULL != anInfo);

    pis = VODI_RESULT_INFO_SPEC(anInfo, plate);
    plate = &pis->pis_plate_variantv[0];

    /* do something */

    return (BoS_NORMAL);
}
```

См. дополнительную информацию в разделах:

- VodiensGet
- VodiresFetchinfo
- VodiResultInfoDestroy
- vodi_plate
- vodi_plate_info_spec
- aorp_error

4 ОПИСАНИЕ ВСПОМОГАТЕЛЬНЫХ УТИЛИТ

4.1 утилита vpwfetch

4.1.1 Обзор

Консольное приложение **vpwfetch** позволяет прочитать автомобильные номера на изображениях.
Программа выполняет следующую последовательность действий:

1. Печатает header-line в header-file.
2. Для каждого изображения:
   (a) печатает prefix-line в prefix-file;
   (b) печатает output-line в output-file (для каждого результата распознавания на текущем изображении);
   (c) печатает sufix-line в sufix-file.
3. Печатает footer-line в footer-file.

Команды:

1. vpwfetch -h
2. vpwfetch [параметры] --config файл-настроек
3. vpwfetch [параметры] входной-файл [...]
4. vpwfetch [параметры] --if входной-файл
5. vpwfetch [параметры] --input-file входной-файл
6. vpwfetch [параметры] --of выходной-файл входной-файл
7. vpwfetch [параметры] --output-file выходной-файл входной-файл
8. vpwfetch [параметры] -S [имя-модуля[, имя-модуля ...]] входной-файл
9. vpwfetch [параметры] --requireds [имя-модуля[, имя-модуля ...]] входной-файл
10. vpwfetch [параметры] -e 'страна[, страна ...]' входной-файл
11. vpwfetch [параметры] --templates 'страна[, страна ...]' входной-файл

4.1.2 Семантика выражений

Перед тем, как применить параметр, утилита **vpwfetch** производит его вычисление. Так как каждое значение параметра интерпретируется как выражение, перед применением **vpwfetch** упрощает эти выражения в строки (либо к тому виду, в котором последующее упрощение невозможно).

Таблица 2: Преобразование выражений

Выражение	**Описание**
[[пустое-выражение]] => пустая-строка	Пустое выражение интерпретируется как пустая строка.
[["""строка"""]] => строка	Строка, трижды взятая в двойные кавычки, интерпретируется как есть.
[['''строка''']] => строка	Строка, трижды взятая в одинарные кавычки, интерпретируется как есть.
[["строка"]] => unescape(строка)	Строка в двойных кавычках интерпретируется как строка с escape-последовательностями.
[['строка']] => unescape(строка)	Строка в одинарных кавычках интерпретируется как строка с escape-последовательностями.
[[слово]] => слово	Слово (последовательность букв или цифр) интерпретируется как строка.
[[(выражение)]] => [[выражение]]	Выражение в скобках интерпретируется как выражение.
[[выражение1 выражение2]] => concat([[выражение1]], [[выражение2]])	Пара выражений интерпретируется как контенация строк, полученных в результате их обработки.
[[$выражение]] => lookup([[выражение]])	Обрабатывается выражение и после этого производится поиск значения среди параметров или в окружении процесса. Если соответствующего параметра не найдено (в т.ч. в окружении), возвращается пустая строка.
[[выражение1 == выражение2]] => strequ([[выражение1]], [[выражение2]])	Обрабатываются выражения и сравниваются их строки. Если полученные строки одинаковые, возвращается слово true, в противном случае — false.
[[выражение1 != выражение2]] => strneq([[выражение1]], [[выражение2]])	Обрабатываются выражения и сравниваются их строки. Если полученные строки одинаковые, возвращается слово false, в противном случае — true.
[[выражение1 && выражение2]] => and([[выражение1]], [[выражение2]])	Обрабатывается выражение1. Если получено значение true, возвращается полученное значение выражения выражение2. В противном случае вернется слово false.
[[выражение1 || выражение2]] => or([[выражение1]], [[выражение2]])	Обрабатывается выражение1. Если получено значение false, возвращается полученное значение выражения выражение2. В противном случае вернется слово true.
[[!выражение]] => not([[выражение]])	Обрабатывается выражение. Если получено значение true, вернется слово false. В другом случае вернется слово true.

Выражение	Описание
[[cond(выражение1 [, ...])]]	Обрабатывается выражение1. Если получено значение true, возвращается выражение2. В другом случае произойдет повтор для выражение3 и выражение4 (и т.д.). Если все пары были проверены и последнее выражение осталось без пары, вернется последнее выражение. В другом случае вернется пустая строка.

Таблица 3: Переменные программы

Переменная	Описание
${src-path}	Путь к текущему изображению.
${src-dirname}	Путь к директории текущего изображения.
${src-basename}	Полное имя текущего изображения.
${src-ext}	Расширение текущего изображения.
${src-name}	Имя текущего изображения без расширения.
${src-seqnum}	Порядковый номер текущего изображения.
${src-timestamp}	Время текущего изображения. При обработке видео — время кадра[1].
${plate-string}	Текущий результат распознавания автомобильного номера (тип — строка).
${plate-type}	Строка с маской текущего результата распознавания.
${plate-id}	Идентификатор типа номерной пластины (шаблона) текущего результата распознавания.
${plate-country-id}	Идентификатор страны, в которой был издан номер (согласно ISO 3166).
${plate-validity}	Числовая оценка возможной корректности текущего результата распознавания[2].
${plate-variant-count}	Количество вариантов текущего результата распознавания.
${plate-outer-left}	Левая координата внешнего прямоугольника текущей номерной пластины.
${plate-outer-top}	Верхняя координата внешнего прямоугольника текущей номерной пластины.
${plate-outer-right}	Правая координата внешнего прямоугольника текущей номерной пластины.
${plate-outer-bottom}	Нижняя координата внешнего прямоугольника текущей номерной пластины.
${plate-exact-left}	Левая координата построенного прямоугольника текущей номерной пластины.

[1] если в качестве входных файлов подаются изображения, **vpwfetch** последовательно обрабатывает их с интервалом 40 мс (по умолчанию)

[2] определяется по собственным критериям AutoSDK

Переменная	Описание
${plate-exact-top}	Верхняя координата построенного прямоугольника текущей номерной пластины.
${plate-exact-right}	Правая координата построенного прямоугольника текущей номерной пластины.
${plate-exact-bottom}	Нижняя координата построенного прямоугольника текущей номерной пластины.
${plate-symrect-list}	Список геометрических позиций символов на изображении текущего результата распознавания.
${plate-inversed}	Признак инверсности текущей номерной пластины.
${plate-direction}	Направление движения транспортного средства. Переменная имеет смысл только при активном режиме динамики.
${plate-angle}	Угол наклона текущей номерной пластины на изображении.
${plate-speed}	Скорость движения транспортного средства.
${plate-index}	Идентификатор траектории движения транспортного средства.
${plate-flags}	Флаги текущего результата распознавания (отображает поле pis_flags структуры результата распознавания vodi_plate_info_spec).
${plate-duplicate}	Признак повторно выданного результата. Имеет смысл только при включенном режиме с повторными выдачами.
${plate-lost}	Признак завершенной траектории движения транспортного средства (новые данные о номерной пластине больше не поступают). Переменная имеет смысл только в при активном режиме динамики.
${plate-invalid}	Признак “невалидной” траектории движения транспортного средства, когда алгоритм трекинга определил эту траекторию как продолжение другой траектории (и она больше не является признаком появления нового автомобиля в кадре). Переменная plate-invalid имеет смысл только в при активном режиме динамики.
${plate-best-frame-timestamp}	Время, когда было получено лучшее изображение[3] номерной пластины (в интервале между ее появлением в кадре и до текущего момента).
${plate-create-timestamp}	Время начала траектории движения (въезда в кадр) транспортного средства.

[3] определяется по собственным критериям AutoSDK

Переменная	Описание
${plate-lost-timestamp}	Время выезда транспортного средства из кадра (номера больше нет в кадре).
${plate-seqnum}	Порядковый номер лучшего изображения номерной пластины от начала траектории до настоящего момента.
${@}	Файл, в который производится запись.
${fps}	Текущее значение скорости анализа изображений.
${avgfps}	Среднее значение скорости анализа изображений за весь период.
${process-duration}	Интервал времени анализа текущего изображения (в микросекундах).

4.1.3 Настройка

Таблица 4: Настройка входящих данных

--encoding, --input-encoding кодировка	Кодировка, в которой передаются параметры. По умолчанию считается, что параметры передаются в кодировке UTF-8.
--if, --input-file файл	Путь к файлу или файлам с изображениями, на которых необходимо распознать автомобильные номера[4]. Примеры: • --if /example.bmp — путь к одному из файлов с изображением; • --if "/*.bmp" — путь к файлам с расширением .bmp; • --if /example.avi — путь к одному видеофайлу, из которого возможно прочитать несколько изображений; • --if "/*.avi" — путь к видеофайлам с расширением .avi.

[4] Следует заметить, что **AutoSDK** как группа библиотек принимает только 8-битовое серое RAW-изображение. При этом **vpwfetch** как утилита поддерживает распространенные графические форматы (например, jpeg, bmp, png и другие).

-u, --raw [ширина, высота[, ориентация]]	Признак того, что входящие изображения имеют формат "RAW". • ширина — ширина изображения; • высота — высота изображения; • ориентация — ориентация изображения: TL (Top Left) или BL (Bottom Left). По умолчанию считается, что изображение имеет ориентацию BL.
-S, --requireds имя | [имя1, ..., имяN] (N >= 0)	Список AORP-модулей для загрузки. Модули будут загружаться в том же порядке, в котором указаны.
--sp, --search-path путь | [путь1, ..., путьN] (N >= 0)	Список путей поиска AORP-модулей.

Таблица 5: Настройка выходных данных

--output-encoding кодировка	Необходимая кодировка выходного файла. По умолчанию считается, что необходимо кодировать выходной файл в UTF-8. Если **vpwfetch** запускается под управлением ОС Windows, то по умолчанию считается, что выходной файл — консоль (тогда необходимо кодировать в CP866 или CP1251). Выражение: $\mathrm{cond}(\mathrm{isatty}(\${@}),\ \text{"CP866"},\ \text{"CP1251"})$.
--hf, --header-file файл	Путь к заголовочному файлу. В этот файл записывается строка, заданная параметром header-line, перед началом анализа изображений. По умолчанию считается, что заголовочный файл совпадает с output-file.
--hl, --header-line строка	Строка, которая будет записана в заголовочный файл. По умолчанию считается, что будет записана пустая строка.
--pf, --prefix-file файл	Путь к префиксному файлу. В этот файл записывается строка, заданная параметром prefix-line, перед началом анализа каждого изображения. По умолчанию считается, что префиксный файл совпадает с output-file.

--pl, --prefix-line строка	Строка, которая будет записана в префиксный файл. По умолчанию считается, что будет напечатана следующая строка: ${src-path}"(${src-seqnum}, "${fps}"):\n".
--of, --output-file файл	Путь к выходному файлу. В этот файл записывается строка, заданная параметром output-line, для каждого полученного результата анализа изображения. По умолчанию считается, что выходной файл будет stdout.
--ol, --output-line строка	Строка будет напечатана в выходной файл. По умолчанию считается, что будет записана следующая строка: ${"plate-string"}"\n".
--sf, --sufix-file файл	Путь к суфиксному файлу. В этот файл записывается строка, заданная параметром sufix-line, после анализа каждого изображения. По умолчанию считается, что суфиксный файл совпадает с output-file.
--sl, --sufix-line строка	Строка, которая будет напечатана в суфиксный файл. По умолчанию считается, что будет записана пустая строка.
--ff, --footer-file файл	Путь к конечному файлу. В этот файл записывается строка, заданная параметром footer-line, в конце анализа изображения. По умолчанию считается, что конечный файл совпадает с output-file.
--fl, --footer-line строка	Строка, которая будет записана в конечный файл. По умолчанию считается, что будет записана пустая строка.

Таблица 6: Настройка детектора номерных пластин

-M, --plate-size-max число	Максимальный размер номерной пластины на изображении в пикселях. По умолчанию — 200.
-m, --plate-size-min число	Минимальный размер номерной пластины на изображении в пикселях. По умолчанию — 45.
-t, --threshold число	Базовый уровень бинаризации изображения. По умолчанию — 21.
-X, --image-scale-factor [X-фактор, Y-фактор]	Параметр уменьшения изображения перед поиском на нем номеров. По умолчанию параметр не задан.

-Z, --image-size [ширина, висота]	Размер изображения перед поиском на нем номеров. Оригинальное изображения будет приведено к этому размеру. По умолчанию размер не задан.
-c, --plate-filter-charc число	Минимально необходимое количество символов на номерной пластине (найденное по простому алгоритму), чтобы продолжить ее дальнейший анализ. По умолчанию — 0.

Таблица 7: Настройка сегментатора

-b, --blur число	Размер маски для алгоритма бинаризации по блюру. По умолчанию — 13.

Таблица 8: Настройки динамики

-y, --with-dynamic логическое-значение	Включение алгоритма динамики. В этом режиме происходит наблюдение за движением автомобилей. По умолчанию алгоритм выключен.
-L, --comparable-time-max интервал-времени	Интервал времени, в рамках которого разрешено проводить сравнение характеристик автомобильных номеров на предмет их принадлежности одному и тому же транспортному средству. По умолчанию — 80 миллисекунд.
-f, --comparable-symbols-min число	Минимальный процент совпавших символов двух номерных пластин, чтобы их отнести к одному и тому же автомобилю. По умолчанию — 80 процентов.
--output-timeout интервал-времени	Интервал от начала наблюдения за движением автомобиля до первой выдачи результата пользователю. По умолчанию — 2 секунды.
--with-duplicate логическое-значение	Включение режима периодической выдачи результатов наблюдения за движением автомобиля. По умолчанию алгоритм выключен.
--output-period интервал-времени	Период повторной выдачи результата наблюдения за движением автомобиля. Параметр может использоваться только при включенном параметре периодической выдачи результата (--with-dublicate). По умолчанию — 500 миллисекунд.
--duration-without-access интервал-времени	Интервал времени с момента завершения наблюдения за движением автомобиля до его последней выдачи пользователю. По прошествии этого интервала считается, что автомобиль уехал, и наблюдение за ним больше не проводится. По умолчанию — 3 секунды.

Таблица 9: Общие настройки

--config файл	Загрузка конфигурационного файла.
-k, --thread-max число	Максимальное количество потоков параллельного видеоанализа. По умолчанию — 1.
-s, --plate-star-max число	Максимально разрешенное количество нераспознанных символов на номерной пластине, при котором результат еще можно вернуть пользователю. По умолчанию — 0.
-p, --plate-validity-min число	Минимально разрешенная достоверность результата распознавания, при которой этот результат еще может быть выдан пользователю. По умолчанию — 0.
-e, --templates "страна [страна ...]"	Задает список шаблонов номеров, с которыми будет происходить сравнение в процессе распознавания. По умолчанию — * (все доступные шаблоны). Используйте программу lsvpwi, чтобы напечатать информацию о шаблонах номеров (см. Утилита lsvpwi), а также программу lsvpwc, чтобы узнать, какие шаблоны номеров могут использоваться согласно текущей лицензии (см. Утилита lsvpwc). Кроме того, ознакомьтесь с грамматикой задания множества шаблонов (см. описание грамматики для обозначения множества шаблонов номерных пластин), которая используется этими программами, а также данной опцией vpwfetch.
-h	Напечатать отчет и завершить работу.

4.1.4 Завершение работы и примеры команд

В случае успешного завершения работы возвращается код со значением 0. В другом случае программа возвращает код со значением 1 и передает сообщение об ошибке в strerr.

Таблица 10: Примеры использования **vpwfetch**

vpwfetch -S "vpwi-co" foo.bmp	Прочитать колумбийские номера из файла foo.bmp.
vpwfetch -S "vpwi-co" *.bmp	Прочитать колумбийские номера из множества файлов *.bmp.
vpwfetch -S "vpwi-co" -y 1 *.bmp	Прочитать колумбийские автомобильные номера из множества файлов *.bmp в режиме "Динамика".

4.2 утилита lsvpwi

4.2.1 Обзор

Консольное приложение **lsvpwi** печатает информацию о шаблонах номерных пластин, которые поддерживает используемая версия **AutoSDK**.

4.2.2 Опции и примеры

> Перед тем, как использовать утилиту, рекомендуется создать переменную окружения AORP_MODULE_PATH, значением которой должен быть путь к директории с модулями стран (vpwi-[код_страны].dll). По отношению к директории, в которой находится утилита, модули стран обычно расположены в ..\libexec\aorp\modules.

Если переменная окружения не создана, необходимо явно указывать путь аргументом опций -m или --mp при каждом вызове приложения (см. таблицу ниже).

Таблица 11: Опции **lsvpwi**

Опция	**Действие**
--help	Напечатать список команд и их значения.
-m --modules-path --mp FilePath	Указать путь поиска модулей стран.
-h --human-readable --hr	Напечатать буквенные коды стран (согласно ISO 3166) в составе идентификаторов шаблонов, а не числовые. Например, вместо идентификатора 170.[номер_шаблона], при использовании данной опции будет выведен идентификатор co.[номер_шаблона]: таким образом будет легче определить, что шаблоны пренадлежат Колумбии.
-r --react	Выводить сообщения об ошибках выполнения (например, когда шаблоны указанной страны не поддерживаются).
-f --force	Игнорировать сообщения об ошибках выполнения (опция активирована по умолчанию).
-o --output-file --of FilePath --stdout	Вывести информацию о шаблонах на консоль либо записать в файл по пути FilePath (по умолчанию вывод в stdout).

-s --output-fields --fields Fields	Указать названия полей, которые необходимо показать для запрашиваемой группы шаблонов (по умолчанию выводится информация по стандартным полям). Названия полей вводятся через запятую (см. примеры). Доступные поля: • id — новый идентификатор шаблона. • oldid — устаревший идентификатор шаблона. • mask — маска шаблона (Z — буква, X — число). • linec — количество строк номера на пластине. • symc — количество символов номера на пластине. • print_format — формат печати распознанного номера в строку (порядок вывода символов).[5] • real_size — размер номерной пластины, мм (согласно государственным стандартам). • background, foreground — цвет фона и символов номерной пластины соответственно.[6] • properties — зарезервировано.
-t --output-format --format Format	Выбрать формат вывода информации. Доступные форматы: 'table' (по умолчанию), 'json', 'cut' (CSV-формат).

Если не специфицировать один или больше кодов или названий стран (на английском), шаблоны номеров которых необходимо отобразить, по умолчанию будет отображен список *всех* поддерживаемых шаблонов.

Таблица 12: Примеры использования **lsvpwi**

lsvpwi nl de fr --hr -s id,oldid	Показать идентификаторы (новые и устаревшие) для шаблонов Нидерландов, Германии и Франции; при этом идентификаторы нужно показать с буквенными кодами стран.

[5] Формат печати построен на основе позиционных переменных $n, где n — порядковый номер символа. Печать номера задается самим шалоном (как правило, слева направо, сверху вниз). Отсутствие значения у данного параметра означает печать в том порядке, в котором получается строка при распознавании. Также данный параметр может специфицировать непечатаемые/нераспознаваемые символы (например, дефис).

[6] Константы определены в <Vodi/clrsType.h>.

lsvpwi at -s "id, print_format, linec, symc"	Показать формат печати, количество строк/символов каждого шаблона Австрии.
lsvpwi Colombia --output-file Colombian_templates.txt --output-format json	Записать в файл Colombian_templates.txt информацию о шаблонах Колумбии в формате JSON.

4.3 утилита lsvpwc

4.3.1 Обзор

Консольное приложение **lsvpwc** сканирует локальную машину на предмет доступных лицензионных ключей Sentinel HASP и, если поиск дал результаты, выводит информацию о них.

4.3.2 Опции

> Перед тем, как использовать утилиту, рекомендуется создать переменную окружения AORP_MODULE_PATH, значением которой должен быть путь к директории с модулями стран (vpwi-[код_страны].dll). По отношению к директории, в которой находится утилита, модули стран обычно расположены в ..\libexec\aorp\modules.

Если переменная окружения не создана, необходимо явно указывать путь аргументом опций -m или --mp при каждом вызове приложения (см. таблицу ниже).

Таблица 13: Опции **lsvpwc**

Опция	**Действие**
--help	Напечатать список команд и их значения.
-m --modules-path --mp FilePath	Указать путь поиска модулей стран.
-o --output-file --of FilePath --stdout	Вывести информацию о лицензии на консоль либо записать в файл по пути FilePath (по умолчанию вывод в stdout).
-t --output-format --format Format	Выбрать формат вывода информации. Доступные форматы: 'table' (по умолчанию), 'json', 'cut' (CSV-формат).
-g --group	Суммировать и вывести возможности, предоставляемые всеми ключами защиты, которые доступны на локальной машине.

Пример вывода программы:

```
1  $lsvpwc
2  +----------+----------+-------+------------+-----------+
3  |    key_id|key_series|fps_max|nthreads_max|  templates|
4  +----------+----------+-------+------------+-----------+
5  |1160193448|    107392|     25|           4|ru kz uz kg|
```

```
|1528129579|    106763|     25|          16|   ua ru su|
+----------+----------+-------+------------+-----------+

$lsvpwc -g
+------+----------+-------+------------+-----------------+
|key_id|key_series|fps_max|nthreads_max|        templates|
+------+----------+-------+------------+-----------------+
|  ----|      ----|     25|          20|kz kg ru ua su uz|
+------+----------+-------+------------+-----------------+
```

Параметры лицензии:

- key_id — уникальный идентификатор лицензионного ключа.
- key_series — уникальный идентификатор компании, которая предоставила вам данную лицензию. [7]
- fps_max — максимальное количество кадров в секунду, которое может быть обработано движком **AutoSDK** согласно данному лицензионному ключу.
- templates — список стран, шаблоны номеров которых будут доступны для использоваться в распознавании.
- flags — дополнительные параметры лицензионного ключа:
  - D — (output-delay) выводится для лицензий, которые модифицируют работу **AutoSDK** таким образом, что результаты распознавания отдаются с задержкой 30 секунд.

[7] Компания Gemalto (производитель лицензионных ключей и ПО для защиты интеллектуальной собственности) присваивает этот идентификатор (англ. *batch code*) каждой компании-клиенту, которая пользуется ее решениями для защиты своих продуктов. Идентификаторы компании VIT: EAOWT и AAOTB (числовые эквиваленты — 107392 и 106763).

5 ОПИСАНИЕ ОПЕРАЦИЙ С AORP-ОБЪЕКТАМИ

5.1 vpwprincopen

5.1.1 Имя

VpwprincOpen — открыть/создать принципал.

5.1.2 Синтаксис

```
#include <Vodi/objects/Vpwprinc.h>

aorp_object_t
VpwprincOpen(
    aorp_opflags_t Flags /* = 0 */,
    bo_pointer_t aMemory /* = NULL */,
    struct vodi_vpw_princ_param *aParm,
    struct aorp_error *anErrPtr /* = NULL */
    );
```

5.1.3 Описание

- Функция **VpwprincOpen** открывает принципал и возвращает на него указатель.
- Аргумент **Flags** дает возможность управлять процессом конструирования объекта. По умолчанию передается 0.
- С помощью аргумента **aMemory** можно передать указатель на память, которая будет выделена для создания объекта. По умолчанию можно передавать NULL. В этом случае конструктор объекта сам выделит память под объект.
- Через аргумент **aParm** можно передать начальные параметры работы принципала. Дополнительная информация содержится в разделе VodiprincSetparam.

Таблица 14: Поля структуры **vodi_vpw_princ_param**

Поле	**Описание**
enable	Эквивалент VodiCTL_VPW_PRINCIPAL_ENABLE.
thread_max	Максимально количество потоков распознавания (см. далее), которые при необходимости создаются в процессе работы. При значении 0, процесс распознавания будет происходить в том же потоке который его запускает. Не рекомендуется задавать значение, превышающее количество ядер процессора (с учетом Hyper Threading).
image_width	Эквивалент VodiCTL_VPW_IMAGE_WIDTH.
image_height	Эквивалент VodiCTL_VPW_IMAGE_HEIGHT.
image_brightnes	Эквивалент VodiCTL_VPW_IMAGE_BRIGHTNES.
image_contrast	Эквивалент VodiCTL_VPW_IMAGE_CONTRAST.
image_blur	Эквивалент VodiCTL_VPW_IMAGE_BLUR.
image_treshold	Эквивалент VodiCTL_VPW_IMAGE_THRESHOLD.
image_reversed	Не используется (deprecated). Должен быть установлен в 0.
image_angle	Эквивалент VodiCTL_VPW_IMAGE_ANGLE.
analysed_zone	Эквивалент VodiCTL_VPW_IMAGE_ANALYSE_ZONE.
analysed_zone_count	Актуальное количество элементов в массиве analysed_zone.
plate_size_max	Эквивалент VodiCTL_VPW_PLATE_SIZE_MAX.
plate_size_min	Эквивалент VodiCTL_VPW_PLATE_SIZE_MIN.
plate_inverse_analyse	Эквивалент VodiCTL_VPW_PLATE_INVERSE_ANALYSE.
plate_probability_min	Эквивалент VodiCTL_VPW_PLATE_PROBABILITY_MIN.
plate_star_max	Эквивалент VodiCTL_VPW_PLATE_STAR_MAX.
md_enable	Не используется. Должен быть равен 0.
md_cell_size	Не используется. Должен быть равен 0.
md_threshold	Не используется. Должен быть равен 0.
md_square_min	Не используется. Должен быть равен 0.
md_mask	Не используется. Должен быть равен 0.
md_mask_h_size	Не используется. Должен быть равен 0.
md_mask_v_size	Не используется. Должен быть равен 0.
dynamic_enable	Эквивалент VodiCTL_VPW_DYNAMIC_ENABLE.
log_settings	Эквивалент VodiCTL_VPW_LOG_SETTINGS

5.1.4 Возвращаемые значения

В случае успеха функция **VpwprincOpen** возвращает указатель на созданный объект. В противном случае — функция возвращает NULL как статус ошибки и записывает её описание в структуру, указанную аргументом **anErrPtr**.

5.2 vpwprincopen_v3

5.2.1 Имя

VpwprincOpen_v3 — открыть/создать принципал.

5.2.2 Синтаксис

```
#include <Vodi/objects/Vpwprinc.h>

aorp_object_t
VpwprincOpen_v3(
    aorp_opflags_t Flags __myc_defval__(0),
    bo_pointer_t aMemory __myc_defval__(NULL),
    unsigned aThreadcMax __myc_defval__(0),
    _Bool (*aLicensePred)(vpw_license_t const *) __myc_defval__(NULLF),
    vpw_license_t const *(*aSelectLicenseOp)(
        vpw_license_t const *, vpw_license_t const *) __myc_defval__(NULLF),
    struct vodi_vpw_princ_param const *aParm __myc_defval__(NULL),
    struct aorp_error *anErrPtr __myc_defval__(NULL)
    );
```

5.2.3 Описание

- Функция **VpwprincOpen_v3** открывает принципал и возвращает на него указатель.
- Аргумент **Flags** дает возможность управлять процессом конструирования объекта. По умолчанию передается 0.
- С помощью аргумента **aMemory** можно передать указатель на память, которая будет выделена для создания объекта. По умолчанию можно передавать NULL. В этом случае конструктор объекта сам выделит память под объект.
- Аргумент **aThreadcMax** задает максимальное количество потоков распознавания, которые при необходимости создаются в процессе работы. При значении 0, процесс распознавания будет происходить в том же потоке который его запускает. Не рекомендуется задавать значение, превышающее количество ядер процессора (с учетом Hyper Threading).
- Аргуменом **aLicensePred** передается указатель на предикат фильтрации доспных лицензий. По умолчанию можно передавать NULLF. В этом случае будет задействован предикат **VpwLicenseNormal**, который выбирает лицензии без функции задержки (архивной лицензии). Также есть стандартный предикат **VpwLicenseWithDelay**, который выбирает лицензии только с функцией задержки (архивные лицензии). При необходимости можно написать свой предикат и передать его в качестве аргумента.
- Аргуменом **aSelectLicenseOp** передается указатель на операцию выбора одной лицензии из двух. С помощью этой операции происходит выбор лицензии из множества доступных после этапа фильтрации с

помощью предиката **aLicensePred**. Операция должна быть коммутативной и ассоциативной. По умолчанию можно передавать NULLF. В этом случае будет задействована операция **VpwLicenseWithMaxFPS**, которая выбирает лицензию с максимальным значением **FPS**. При необходимости можно написать свою операцию и передать ее в качестве аргумента.

- Через аргумент **aParm** можно передать начальные параметры работы принципала. Дополнительная информация содержится в разделе VodiprincSetparam.

5.2.4 Возвращаемые значения

В случае успеха функция **VpwprincOpen_v3** возвращает указатель на созданный объект. В противном случае — функция возвращает NULL как статус ошибки и записывает её описание в структуру, указанную аргументом **anErrPtr**.

5.3 aorpretain

5.3.1 Имя

AorpRetain — захватить AORP-объект.

5.3.2 Синтаксис

```
#include <Bo/services/Ucntl.h>

ssize_t
__attribute__((nonnull(1))
AorpRetain(
    aorp_object_t aThis,
    struct aorp_error *anErrPtr /* = NULL */
    );
```

5.3.3 Описание

Функция **AorpRetain** захватывает AORP-объект, переданный аргументом **aThis**. В результате возвращает получившееся количество ссылок на объект.

5.3.4 Возвращаемые значения

В случае успеха функция **AorpRetain** возвращает значение больше или равно 0. В противном случае функция возвращает отрицательное значение как статус ошибки и записывает её описание в структуру, указанную аргументом **anErrPtr**.

5.4 aorprelease

5.4.1 Имя

AorpRelease — отпустить AORP-объект.

5.4.2 Синтаксис

```
#include <Bo/services/Ucntl.h>

ssize_t
__attribute__((nonnull(1))
AorpRelease(
    aorp_object_t aThis,
    aorp_opflags_t Flags /* = 0 */,
    struct aorp_error *anErrPtr /* = NULL */
    );
```

5.4.3 Описание

Функция **AorpRelease** отпускает AORP-объект, переданный аргументом **aThis**.

5.4.4 Возвращаемые значения

В случае успеха функция **AorpRelease** возвращает оставшееся количество ссылок на объект. Если функция вернула 0, то в результате её работы объект был уничтожен. В противном случае функция возвращает отрицательное значение как статус ошибки и записывает её описание в структуру, указанную аргументом **anErrPtr**.

5.5 vodiprincgetparam

5.5.1 Имя

VodiprincGetparam — получить значение параметра.

5.5.2 Синтаксис

```
#include <Vodi/services/Vodiprinc.h>

bo_status_t
__attribute__((nonnull(1))
VodiprincGetparam(
    aorp_object_t aThis,
    struct aorp_error *anErrPtr /* = NULL */,
    int aParam,
    ...
    );
```

5.5.3 Описание

Функция **VodiprincGetparam** возвращает значение параметра, указанного аргументом **aParam**. В зависимости от типа параметра, в функцию могут передаваться дополнительные указатели, в которые будут записаны актуальные значения параметра.
Дополнительная информация о параметрах содержится в разделе VodiprincSetparam.

5.5.4 Возвращаемые значения

В случае успеха функция **VodiprincGetparam** возвращает актуальное значение параметра, которое больше или равно 0. В противном случае функция возвращает отрицательное значение как статус ошибки и записывает её описание в структуру, указанную аргументом **anErrPtr**.

5.6 vodiprincsetparam

5.6.1 Имя

VodiprincSetparam — установить параметр.

5.6.2 Синтаксис

```
#include <Vodi/services/Vodiprinc.h>

bo_status_t
__attribute__((nonnull(1))
VodiprincSetparam(
    aorp_object_t aThis,
    struct aorp_error *anErrPtr /* = NULL */,
    int aParam,
    ...
    );
```

5.6.3 Описание

Функция **VodiprincSetparam** устанавливает значение параметра, указанного параметром **aParam**. В зависимости от типа параметра, в функцию могут быть переданы дополнительные аргументы для установки значения.

Таблица 15: Параметры VodiprincSetparam

Параметр	**Описание**
VodiCTL_VPW_PRINCIPAL_ENABLE	Не используется (deprecated).
VodiCTL_VPW_PLATE_INVERSE_ANALYSE	Не используется (deprecated).
VodiCTL_VPW_PLATE_PROBABILITY_MIN	Минимальный процент схожести между результатом распознавания и соответствующим шаблоном номерной пластины, при котором этот результат может считаться результатом распознавания номерной пластины. С помощью данного параметра происходит фильтрация получаемых результатов по достоверности. Тип — unsigned.
VodiCTL_VPW_PLATE_STAR_MAX	Максимальное количество нераспознанных символов на номерной пластине, при котором результат еще будет считаться результатом распознавания номерной пластины. Тип — unsigned.

Параметр	Описание
VodiCTL_VPW_PLATE_SIZE_MAX	Максимальная ширина номера (в пикселях), при которой результат еще будет считаться результатом распознавания номерной пластины. Тип — unsigned.
VodiCTL_VPW_PLATE_SIZE_MIN	Минимальная ширина номера (в пикселях), при которой результат еще будет считаться результатом распознавания номерной пластины. Тип — unsigned.
VodiCTL_VPW_PLATE_EXTRA_ANGLE_ANALYSE	Включить/выключить алгоритм учета перспективы изображения номерной пластины. Тип — int(boolean).
VodiCTL_VPW_PLATE_EXTRA_RANGES_ANALYSE	Включить/выключить более тщательный поиск кандидатов в номера. Тип — int(boolean).
VodiCTL_VPW_PLATE_FILTER_ROFACTOR	Коэффициент фильтрации номерных пластин по так называемой плотности изображения — отношению количества белых пикселей к общему количеству пикселей (первая стратегия). Тип — unsigned. Данный коэффициент используется при бинаризации изображения и имеет оптимальные значения, которые определяются разработчиками AutoSDK с помощью собственных тестовых выборок. Параметр считается служебным, и задавать его значение нужно согласно рекомендациям специалистов технической поддержки AutoSDK.
VodiCTL_VPW_ PLATE_FILTER_RODROPFACTOR	Коэффициент фильтрации номерных пластин по плотности изображения (вторая стратегия). Тип — unsigned. Данный коэффициент используется при бинаризации изображения и имеет оптимальные значения, которые определяются разработчиками AutoSDK с помощью собственных тестовых выборок. Параметр считается служебным, и задавать его значение нужно согласно рекомендациям специалистов технической поддержки AutoSDK.

Параметр	Описание
VodiCTL_VPW_ PLATE_IMAGE_SCALE_FACTOR	Коэффициент изменения размера изображения перед поиском на нем номерных пластин (уменьшение изображения может значительно сократить время поиска). Параметр задается в относительных единицах, и представляет собой два масштабирующих коэффициента (для ширины и высоты изображения). Данный параметр взаимоисключаем с параметром VodiCTL_VPW_PLATE_IMAGE_SIZE. Тип — **struct vodi_scale_factor** *. Например, если необходимо уменьшить исходное изображение по ширине в два раза, а по высоте в три, для данного параметра устанавливается значение {2, 3}.
VodiCTL_VPW_PLATE_IMAGE_SIZE	Размер, к которому необходимо привести исходное изображение. В пределах полученного изображения (как правило, уменьшенного) будет проведен поиск номерных пластин. Такое уменьшение изображения позволяет уменьшить нагрузку на алгоритм поиска номеров. Данный параметр взаимоисключаем с параметром VodiCTL_VPW_PLATE_IMAGE_SCALE_FACTOR. Тип — **struct vodi_size** *. Значения задаются в абсолютных единицах.
VodiCTL_VPW_PLATE_TEMPLATE	Включить/выключить шаблон, указанный первым аргументом. Второй аргумент определяет, что нужно сделать (0 — выключить, !0 — включить). Тип первого аргумента — **struct vodi_vpw_country_id** *. Тип второго аргумента — int(boolean).

Параметр	Описание
VodiCTL_VPW_PLATE_FILTER_SYMCOUNT	Включить/выключить алгоритм простого фильтра номерных пластин по минимальному количеству распознанных символов на них. Если алгоритм включен (значение параметра больше 0), производится базовый поиск символов на кандидате в номерную пластину (геометрия, пропорции). Если на кандидате символов определяется меньше, чем указано в значении данного параметра, этот кандидат не считается номерной пластиной. То есть значение данного параметра является минимальным количеством символов, которые должны присутствовать на кандидате в номерную пластину при работе базового алгоритма. Тип — unsigned.
VodiCTL_VPW_PLATE_PRECISE_ANALYSE	Включить/выключить углубленный анализ изображения. Повышает качество распознавания в неблагоприятных условиях (например, если характеристики/настройки камеры не полностью соответствуют требованиям либо при плохой погоде). Увеличивает время обработки кадра на 20-30% в зависимости от его размера. В нормальных условиях данный параметр не влияет на качество распознавания. Тип — int(boolean).
VodiCTL_VPW_DNN_DEVICES	Список устройств на которых вычисляются глубокие нейронные сети. Тип первого аргумента — **vodi_devtyp_t** *. Тип второго аргумента — unsigned.
VodiCTL_VPW_PLATECANDS_METHODS	Множество методов получения кандидатов на номерные пластины. Доступные методы VodiF_VPW_PLATECANDS_BY_MORPH и VodiF_VPW_PLATECANDS_BY_DNN. Тип аргумента — unsigned.
VodiCTL_VPW_ANALYSE_LEVEL	Уровень анализа номерной пластины. Доступны уровни VodiK_VPW_PLATECANDS_ANALYSE, VodiK_VPW_SYMCANDS_ANALYSE и VodiK_VPW_TEXT_ANALYSE. Тип аргумента — int.
VodiCTL_VPW_IMAGE_WIDTH	Не используется (deprecated).

Параметр	Описание
VodiCTL_VPW_IMAGE_HEIGHT	Не используется (deprecated).
VodiCTL_VPW_IMAGE_BRIGHTNES	Не используется (deprecated).
VodiCTL_VPW_IMAGE_CONTRAST	Не используется (deprecated).
VodiCTL_VPW_IMAGE_BLUR	Параметр, используемый для внутренних потребностей. Рекомендуется задавать значение 13. Тип — int.
VodiCTL_VPW_IMAGE_THRESHOLD	Порог бинаризации цветных пикселей в черно-белые. Параметр используется для внутренних потребностей. Рекомендуется задавать значение 21. Тип — unsigned.
VodiCTL_VPW_IMAGE_ANALYSE_ZONE	Массив зон распознавания, где ожидается появление номера. Тип первого аргумента — **struct vodi_rect** *. Тип второго аргумента — unsigned.
VodiCTL_VPW_IMAGE_AZONE_MASK	Изображение-маска области анализа. Тип — **struct vodi_image** *.
VodiCTL_VPW_IMAGE_NOT_ANALYSE_ZONE	Не используется (deprecated).
VodiCTL_VPW_IMAGE_ANGLE	Не используется (deprecated).
VodiCTL_VPW_DYNAMIC_ENABLE	Включить/выключить режим “Динамика”. Тип — int(boolean).
VodiCTL_VPW_DYNAMIC_OUTPUT_TIMEOUT	Минимально необходимое время наблюдения за номерной пластиной (в микросекундах), прежде чем выдать результат распознавания пользователю. Данный параметр может использоваться только при включенном режиме “Динамика”. В этом режиме производится наблюдение за траекторией движения автомобиля, и пользователь не сразу получает результат распознавания номера, а по прошествии времени, указанного для данной настройки. В таком случае первый результат распознавания будет заменен на более качественный и впоследствии выдан пользователю. Если для данного значения установлен параметр 0, пользователю выдается первый результат распознавания найденного номера. Следует учитывать, что по прошествии времени, указанного в данном параметре, наблюдение за траекторией номера продолжается, пока он не скроется из кадра. Тип — bo_usec_t.

Параметр	Описание
VodiCTL_VPW_DYNAMIC_OUTPUT_PERIOD	Период времени (в микросекундах), за который необходимо выдать пользователю результат распознавания. Данный параметр можно использовать только при установленном параметре VodiCTL_VPW_DYNAMIC_WITH_DUPLICATE. Тип — bo_usec_t.
VodiCTL_VPW_DYNAMIC_WITH_DUPLICATE	Включить/выключить периодическую выдачу результатов распознавания номерных пластин. Тип — int(boolean).
VodiCTL_VPW_ DYNAMIC_DURATION_WITHOUT_ACCESS	Максимально допустимое время отсутствия номерной пластины в зоне наблюдения (в микросекундах). По истечении этого времени номерная пластина считается потерянной и выдается пользователю с установленным флажком VodiF_RESULT_LOST. Тип — bo_usec_t.
VodiCTL_VPW_LOG_SETTINGS	Включить/выключить запись в лог-файл всех параметров распознавания. Тип — int(boolean).

5.6.4 Возвращаемые значения

В случае успеха функция **VodiprincSetparam** возвращает значение больше или равно 0. В противном случае функция возвращает отрицательное значение как статус ошибки и заполняет её описание в структуру, указанную аргументом **anErrPtr**.

5.7 vodiprincapplyparam

5.7.1 Имя

VodiprincApplyparam — применить параметры.

5.7.2 Синтаксис

```
#include <Vodi/services/Vodiprinc.h>

bo_status_t
__attribute__((nonnull(1))
VodiprincApplyparam(
    aorp_object_t aThis,
    struct aorp_error *anErrPtr /* = NULL */
    );
```

5.7.3 Описание

Функция **VodiprincApplyparam** применяет параметры, установленные функцией **VodiprincSetparam**.

5.7.4 Возвращаемые значения

В случае успеха функция **VodiprincApplyparam** возвращает значение больше или равно 0. В противном случае функция возвращает отрицательное значение как статус ошибки и заполняет её описание в структуру, указанную аргументом **anErrPtr**.

5.8 vodiprinccontrol

5.8.1 Имя

VodiprincControl — управление принципалом (распознавателем).

5.8.2 Синтаксис

```
# include <Vodi/services/Vodiprinc.h>

bo_status_t
__attribute__((nonnull(1)))
VodiprincControl(
    aorp_object_t aThis,
    struct aorp_error *anErrPtr,
    int aCommand,
    ...
    );
```

5.8.3 Описание

Функция **VodiprincControl** принимает следующие параметры:

- **aThis** — принципал.
- **anErrPtr** — указатель на структуру описания ошибки.
- **aCommand** — команда, которую нужно выполнить.
- **...** — аргументы, специфичные для команды (параметры команды).

Используемые команды:

- **VodiCTL_GETVPWI** — команда вернуть множество шаблонов, которые используются на данный момент (ожидается параметр char ** — указатель на строку, куда будет записано текущее множество шаблонов).
- **VodiCTL_ADDVPWI** — команда добавить указанное множество шаблонов к текущему (ожидается параметр char const * — добавляемое множество шаблонов).
- **VodiCTL_SETVPWI** — команда сбросить выбранное на данный момент множество шаблонов, заменив их на множество, переданное параметром (ожидается char const * — устанавливаемое множество шаблонов).
- **VodiCTL_DELVPWI** — команда удалить из текущего множества шаблонов те, что заданы параметром (ожидается char const * — удаляемое множество шаблонов).

5.8.4 Возвращаемые значения

В случае успеха функция **VodiprincControl** возвращает значение больше или равно 0. В противном случае функция возвращает отрицательное значение как статус ошибки и записывает её описание в структуру, указанную аргументом **anErrPtr**.

> См. дополнительную информацию в разделах:
>
> - описание функций для работы с шаблонами
> - описание грамматики для обозначения множества шаблонов номерных пластин

5.9 vodiprincprocess

5.9.1 Имя

VodiprincProcess — провести анализ изображения.

5.9.2 Синтаксис

```
#include <Vodi/services/Vodiprinc.h>

bo_status_t
__attribute__((nonnull(1,2,7))
VodiprincProcess(
    aorp_object_t aThis,
    struct vodi_image *anImage,
    size_t anAnalizeZonec /* = 0 */,
    vodi_rect_t const anAnalizeZonev[] /* = NULL */,
    struct vodi_ucontext const *anUserCtx /* = NULL */,
    aorp_object_t anInputEnsemble /* = NULL */,
    aorp_object_t *anOutputEnsemblePtr,
    void *Options /* = NULL */,
    struct aorp_error *anErrPtr /* = NULL */
    );
```

5.9.3 Описание

- Функция **VodiprincProcess** анализирует изображение, которое было передано параметром **anImage**.
- Можно указать прямоугольные области на изображении (зоны распознавания), где функция будет проводить анализ. Количество таких областей задается аргументом **anAnalizeZonec**, а указатель на подготовленный массив прямоугольников — аргументом **anAnalizeZonev**. Если передать в качестве количества областей значение 0, аргумент **anAnalizeZonev** игнорируется, и анализ будет проводиться по всему изображению.
- Через аргумент **anUserCtx** можно передать контекст пользователя, который будет связываться с каждым результатом анализа. Детальную информацию о контексте пользователя можно найти в разделе vodi_ucontext.
- Каждый результат анализа может быть занесен в заранее подготовленный ансамбль, который передается с помощью аргумента **anInputEnsemble**. Если этот параметр равен NULL, для занесения результатов будет создан новый ансамбль. В целом, на выходной (результирующий) ансамбль указывет параметр **anOutputEnsemblePtr**.
- Для анализа/поиска автомобильных номеров можно передать дополнительные параметры аргументом **Options**. В данном случае указатель должен указывать на **struct vodi_vpw_options**.

5.9.4 Возвращаемые значения

В случае успеха функция **VodiprincProcess** возвращает значение больше или равно 0. В противном случае функция возвращает отрицательное значение как статус ошибки и записывает её описание в структуру, указанную аргументом **anErrPtr**.

5.10 vodiprincflush

5.10.1 Имя

VodiprincFlush — очистить внутренние буферы.

5.10.2 Синтаксис

```
#include <Vodi/services/Vodiprinc.h>

bo_status_t
__attribute__((nonnull(1,3))
VodiprincFlush(
    aorp_object_t aThis,
    aorp_object_t anInputEnsemble /* = NULL */,
    aorp_object_t *anOutputEnsemblePtr,
    struct aorp_error *anErrPtr /* = NULL */
    );
```

5.10.3 Описание

Функция **VodiprincFlush** очищает внутренние буферы результатов анализа, полученных в режиме "Динамика".
Каждый результат из буфера заносится в ансамбль, который можно передать аргументом **anInputEnsemble**. Если **anInputEnsemble** равен NULL, для занесения результатов будет создан новый ансамбль.
Ансамбль с результатами передается выходным аргументом **anOutputEnsemblePtr**.

5.10.4 Возвращаемые значения

В случае успеха функция **VodiprincFlush** возвращает значение больше или равно 0. В противном случае функция возвращает отрицательное значение как статус ошибки и записывает её описание в структуру, указанную аргументом **anErrPtr**.

5.11 vodienscount

5.11.1 Имя

VodiensCount — вернуть количество результатов в ансамбле.

5.11.2 Синтаксис

```
#include <Vodi/services/Vodiens.h>

ssize_t
__attribute__((nonnull(1))
VodiensCount(
    aorp_object_t aThis,
    struct aorp_error *anErrPtr /* = NULL */
    );
```

5.11.3 Возвращаемые значения

В случае успеха функция **VodiensCount** возвращает количество результатов анализа в ансамбле. В противном случае функция возвращает отрицательное значение как статус ошибки и записывает её описание в структуру, указанную аргументом **anErrPtr**.

5.12 vodiensclear

5.12.1 Имя

VodiensClear — очистить ансамбль результатов.

5.12.2 Синтаксис

```
#include <Vodi/services/Vodiens.h>

bo_status_t
__attribute__((nonnull(1))
VodiensClear(
    aorp_object_t aThis,
    struct aorp_error *anErrPtr /* = NULL */
    );
```

5.12.3 Описание

Функция **VodiensClear** очищает ансамбль, передаваемый аргументом **aThis**. При этом для всех результатов в ансамбле применяется функция **AorpRelease**.

5.12.4 Возвращаемые значения

В случае успеха функция **VodiensClear** возвращает значение больше или равно 0. В противном случае функция возвращает отрицательное значение как статус ошибки и записывает её описание в структуру, указанную аргументом **anErrPtr**.

5.13 vodiensadd

5.13.1 Имя

VodiensAdd — добавить результат.

5.13.2 Синтаксис

```
#include <Vodi/services/Vodiens.h>

bo_status_t
__attribute__((nonnull(1,2))
VodiensAdd(
    aorp_object_t aThis,
    aorp_object_t aResult,
    struct aorp_error *anErrPtr /* = NULL */
    );
```

5.13.3 Описание

Функция **VodiensAdd** добавляет результат, передаваемый аргументом **aResult**, в ансамбль-аргумент параметра **aThis**. При этом для результата применяется функция **AorpRetain**.

5.13.4 Возвращаемые значения

В случае успеха функция **VodiensAdd** возвращает значение больше или равно 0. В противном случае функция возвращает отрицательное значение как статус ошибки и записывает её описание в структуру, указанную аргументом **anErrPtr**.

5.14 vodiensremove

5.14.1 Имя

VodiensRemove — забирает результат из ансамбля.

5.14.2 Синтаксис

```
#include <Vodi/services/Vodiens.h>

bo_status_t
__attribute__((nonnull(1,3))
VodiensRemove(
    aorp_object_t aThis,
    bo_index_t anIndex,
    aorp_object_t *aResultPtr,
    struct aorp_error *anErrPtr /* = NULL */
    );
```

5.14.3 Описание

С помощью аргумента **anIndex** указывается порядковый номер того результата в ансамбле (нумерация начинается с 0), который нужно забрать. Если передать отрицательное значение, результат будет выбран с конца ансамбля. Забранный результат записывается в выходной аргумент **aResultPtr**. К забранному результату не применяются функции **AorpRetain**/**AorpRelease**.

5.14.4 Возвращаемые значения

В случае успеха функция **VodiensRemove** возвращает значение больше или равно 0. В противном случае функция возвращает отрицательное значение как статус ошибки и записывает её описание в структуру, указанную аргументом **anErrPtr**.

5.15 vodiensdelete

5.15.1 Имя

VodiensDelete — удалить результат.

5.15.2 Синтаксис

```
#include <Vodi/services/Vodiens.h>

bo_status_t
__attribute__((nonnull(1))
VodiensDelete(
    aorp_object_t aThis,
    bo_index_t anIndex,
    struct aorp_error *anErrPtr /* = NULL */
    );
```

5.15.3 Описание

В аргументе **anIndex** указывается порядковый номер результата в ансамбле (нумерация начинается с 0), который нужно удалить. Если передать отрицательное значение, результат будет выбран с конца ансамбля. К выбранному результату будет применена функция **AorpRelease**.

5.15.4 Возвращаемые значения

В случае успеха функция **VodiensDelete** возвращает значение больше или равно 0. В противном случае функция возвращает отрицательное значение как статус ошибки и записывает её описание в структуру, указанную аргументом **anErrPtr**.

5.16 vodiensget

5.16.1 Имя

VodiensGet — получить результаты.

5.16.2 Синтаксис

```
#include <Vodi/services/Vodiens.h>

ssize_t
__attribute__((nonnull(1))
VodiensGet(
    aorp_object_t aThis,
    bo_index_t aFrom,
    bo_index_t aTo,
    size_t aCount,
    aorp_object_t aResultv[],
    struct aorp_error *anErrPtr /* = NULL */
    );
```

5.16.3 Описание

- Функция **VodiensGet** забирает результаты из ансамбля, передаваемого аргументом **aThis**. Результаты выбираются в диапазоне, указанном индексами [**aFrom**, **aTo**]. Если индекс имеет отрицательное значение, он указывает на элемент в конце ансамбля.
- С помощью аргумента **aCount** указывается максимальное количество результатов, которые можно записать в буфер **aResultv**. Если выбирается больше результатов, чем можно записать в буфер, это не приводит к ошибке.
- К выбранным результатам не применяются функции **AorpRetain**/**AorpRelease**.

5.16.4 Возвращаемые значения

В случае успеха функция **VodiensGet** возвращает выбранное количество результатов (может быть больше, чем указано в аргументе **aCount**). В противном случае функция возвращает отрицательное значение как статус ошибки и записывает её описание в структуру, указанную аргументом **anErrPtr**.

5.17 vodiensunique

5.17.1 Имя

VodiensUnique — оставить уникальные результаты.

5.17.2 Синтаксис

```
#include <Vodi/services/Vodiens.h>

ssize_t
__attribute__((nonnull(1,2))
VodiensUnique(
    aorp_object_t aThis,
    aorp_object_t aPrincipal,
    struct aorp_error *anErrPtr /* = NULL */
    );
```

5.17.3 Описание

Функция **VodiensUnique** оставляет только уникальные результаты в ансамбле, который передается аргументом **aThis**. Аргумент **aPrincipal** должен указывать на принципал, с помощью которого будет производиться данная операция.

5.17.4 Возвращаемые значения

В случае успеха функция **VodiensUnique** возвращает количество оставшихся результатов. В противном случае функция возвращает отрицательное значение как статус ошибки и записывает её описание в структуру, указанную аргументом **anErrPtr**.

5.18 vodienscombine

5.18.1 Имя

VodiensCombine — объединить результаты.

5.18.2 Синтаксис

```
#include <Vodi/services/Vodiens.h>

ssize_t
__attribute__((nonnull(1,2))
VodiensCombine(
    aorp_object_t aThis,
    aorp_object_t aPrincipal,
    int anWithMerge,
    size_t anEnsc,
    aorp_object_t anEnsv[],
    struct aorp_error *anErrPtr /* = NULL */
    );
```

5.18.3 Описание

- Функция **VodiensCombine** объединяет ансамбли в один результирующий, указанный с помощью аргумента **aThis**.
- Аргумент **aPrincipal** должен указывать на принципал, с помощью которого будет производиться данная операция.
- Количество и массив ансамблей, которые необходимо объединить, задаются аргументами **anEnsc** и **anEnsv** соответственно. Результаты в данных ансамблях будут удалены.
- Если аргумент **anWithMerge** имеет ненулевое значение, функция оставит в результирующем ансамбле только уникальные результаты.

5.18.4 Возвращаемые значения

В случае успеха функция **VodiensCombine** возвращает количество оставшихся результатов. В противном случае функция возвращает отрицательное значение как статус ошибки и заполняет её описание в структуру, указанную аргументом **anErrPtr**.

5.19 vodienscombine_v2

5.19.1 Имя

VodiensCombine_v2 — объединить результаты.

5.19.2 Синтаксис

```
#include <Vodi/services/Vodiens.h>

ssize_t
__attribute__((nonnull(1,2))
VodiensCombine_v2(
    aorp_object_t aThis,
    aorp_object_t aPrincipal,
    aorp_opflags_t Flags,
    size_t anEnsc,
    aorp_object_t anEnsv[],
    struct aorp_error *anErrPtr /* = NULL */
    );
```

5.19.3 Описание

Функция **VodiensCombine_v2** объединяет ансамбли в один результирующий (указанный аргументом **aThis**). Аргумент **aPrincipal** должен указывать на принципал, с помощью которого будет производиться данная операция. Количество и массив ансамблей к объединению задаются аргументами **anEnsc** и **anEnsv** соответственно.
С помощью аргумента **Flags** в операцию могут передаваться следующие флажки:

- VodiensF_COMBINE_WITH_MERGE — оставить только уникальные результаты в результирующем ансамбле.
- VodiensF_COMBINE_WITH_DUP — не удалять результаты из ансамблей, передаваемых аргументом **anEnsv**, а передавать их копии в результирующий ансамбль.

5.19.4 Возвращаемые значения

В случае успеха функция **VodiensCombine_v2** возвращает количество результатов в результирующем ансамбле. В противном случае функция возвращает отрицательное значение как статус ошибки и записывает её описание в структуру, указанную аргументом **anErrPtr**.

5.20 vodiresgetuserdata

5.20.1 Имя

VodiresGetuserdata — получить контекст пользователя.

5.20.2 Синтаксис

```
#include <Vodi/services/Vodires.h>

bo_status_t
__attribute__((nonnull(1,2))
VodiresGetuserdata(
    aorp_object_t aThis,
    struct vodi_ucontext *anUserCtx,
    struct aorp_error *anErrPtr /* = NULL */
    );
```

5.20.3 Описание

Функция **VodiresGetuserdata** возвращает контекст пользователя, записывая его по указателю **anUserCtx**. К контексту пользователя не применяется операция **VODI_UCONTEXT_DRetain**.

5.20.4 Возвращаемые значения

В случае успеха функция **VodiresGetuserdata** возвращает значение больше или равно 0. В противном случае функция возвращает отрицательное значение как статус ошибки и заполняет её описание в структуру, указанную аргументом **anErrPtr**.

5.21 vodiressetuserdata

5.21.1 Имя

VodiresSetuserdata — установить контекст пользователя.

5.21.2 Синтаксис

```
#include <Vodi/services/Vodires.h>

bo_status_t
__attribute__((nonnull(1,2))
VodiresSetuserdata(
    aorp_object_t aThis,
    struct vodi_ucontext *anUserCtx,
    struct aorp_error *anErrPtr /* = NULL */
    );
```

5.21.3 Описание

Функция **VodiresSetuserdata** устанавливает контекст пользователя, передаваемый аргументом **anUserCtx**. К контексту пользователя применяется операция **VODI_UCONTEXT_DRetain**.

5.21.4 Возвращаемые значения

В случае успеха функция **VodiresSetuserdata** возвращает значение больше или равно 0. В противном случае функция возвращает отрицательное значение как статус ошибки и записывает её описание в структуру, указанную аргументом **anErrPtr**.

5.22 vodiresfetchinfo

5.22.1 Имя

VodiresFetchinfo — прочитать информацию.

5.22.2 Синтаксис

```
#include <Vodi/services/Vodires.h>

bo_status_t
__attribute__((nonnull(1,2))
VodiresFetchinfo(
    aorp_object_t aThis,
    struct vodi_result_info *anInfo,
    struct aorp_error *anErrPtr /* = NULL */
    );
```

5.22.3 Описание

Функция **VodiresFetchinfo** читает информацию о результате в буфер, указанный аргументом **anInfo**. При этом необходимо установить для поля **ri_type** параметра **anInfo** следующие значения:

- VodiK_VPW_RESULT_INFO — информация о номере. **anInfo** должен указывать на структуру **vodi_vpw_result_info**.
- VodiK_FCW_RESULT_INFO — зарезервировано.

Пользователь должен уничтожить информацию, записанную в буфер (через параметр **anInfo**), вызовом функции **VodiResultInfoDestroy** (см. далее).

5.22.4 Возвращаемые значения

В случае успеха функция **VodiresFetchinfo** возвращает значение больше или равно 0. В противном случае функция возвращает отрицательное значение как статус ошибки и записывает её описание в структуру, указанную аргументом **anErrPtr**.

5.23 vodiresultinfodestroy

5.23.1 Имя

VodiResultInfoDestroy — разрушить информацию о результате.

5.23.2 Синтаксис

```
#include <Vodi/Vodilib.h>

void
__attribute__((nonnull(1))
VodiResultInfoDestroy(
    struct vodi_result_info *anInfo
    );
```

5.23.3 Описание

Функция **VodiResultInfoDestroy** разрушает информацию о результате, указанную аргументом **anInfo**. Используется в паре с функцией **VodiresFetchinfo**.

6 ОПИСАНИЕ СТРУКТУР

6.1 vodi_vpw_country_id

6.1.1 Имя

struct vodi_vpw_country_id — идентификатор шаблона номерной пластины.

6.1.2 Синтаксис

```
#include <Vodi/Vpwtypes.h>

struct vodi_vpw_country_id {
    unsigned int code;
    unsigned int id;
    unsigned int sub_id;
};
```

6.1.3 Описание

- **code** — код страны (согласно ISO 3166-1 numeric).
- **id** — идентификатор шаблона. Допускается значение 0 (предполагает все шаблоны страны, указанной с помощью поля **code**)
- **sub_id** — зарезервирован (требуемое значение — 0).

6.2 vodi_image

6.2.1 Имя

struct vodi_image — изображение.

6.2.2 Синтаксис

```
#include <Vodi/Types.h>

struct vodi_image {
    unsigned      img_flags;
    long          img_width;  /* in pixels */
    long          img_height; /* in pixels */
    unsigned      img_bpp;    /* bits per pixel */
    bo_pointer_t  img_base;
};
```

6.2.3 Описание

- **img_flags** — устанавливать в VodiF_IMAGE_PLAIN (“других флажков нет”), если объект структуры самостоятельно создается пользователем. Если объект создается с помощью функции семейства VodiImage (например, VodiImageCreate), данное поле будет заполнено этой функцией.
- **img_width** — ширина изображения.
- **img_height** — высота изображения.
- **img_bpp** — количество бит на пиксель. Установить в 8.
- **img_base** — указатель на первую строку изображения. Каждая строка изображения должна быть выровнена на 4 байта.

6.3 vodi_vpw_options

6.3.1 Имя

struct vodi_vpw_options — дополнительные параметры анализа.

6.3.2 Синтаксис

```
#include <Vodi/Vpwtypes.h>

struct vodi_vpw_options {
    int          magic;
    vodi_size_t  imgsz;         /* reserved; deprecated */
    u_int32_t    img_seqnum;    /* original image sequence mark */
    bo_utime_t   img_timestamp; /* original image timestamp */
};
```

6.3.3 Описание

- **magic** — должен быть установлен в VodiK_VPW_OPTIONS_WORK_MAGIC. Данный параметр зарезервирован для указания структуры, с которой необходимо работать (на данный момент доступна только vodi_vpw_options).
- **imgsz** — не используется (deprecated).
- **img_seqnum** — порядковый номер изображения в последовательности. Поле используется в режиме “Динамика”.
- **img_timestamp** — временная метка изображения (в микросекундах). Позволяет проводить временной анализ при слежении за номерными пластинами. Используется:
  - в режиме работы “Динамика”;
  - функцией **VodiensUnique**;
  - функцией **VodiensCombine** в режиме “WithMerge”;
  - функцией **VodiensCombine_v2** с флагом VodiensF_COMBINE_WITH_MERGE.

6.4 vodi_ucontext

6.4.1 Имя

struct vodi_ucontext — контекст пользователя.

6.4.2 Синтаксис

```
#include <Vodi/Types.h>

struct vodi_ucontext {
    vodi_uctx_dup_fn       uctx_dup;
    vodi_udata_retain_fn   uctx_dretain;
    vodi_udata_release_fn  uctx_drelease;
    vodi_uctx_change_fn    uctx_change;
    bo_pointer_t           uctx_udata;
};
```

6.4.3 Описание

Структура представляет собой контекст пользователя, который можно передавать в функцию **VodiprincProcess**. Он будет связан с каждым полученным результатом при анализе изображения.

- При связывании контекста с результатом, для контекста вызывается операция, указанная в поле **uctx_dretain**. Если значение этого поля равно NULLF (нулевой указатель на функцию), вызов операции не происходит.
- В процессе работы функции **VodiprincProcess** может возникнуть потребность дублировать имеющийся результат. В этом случае для связанного контекста вызывается операция, указанная в поле **uctx_dup**. Если значение этого поля равно NULLF, вызов операции не происходит.
- Когда результат уничтожается, для связанного с ним контекста вызывается операция, указанная в поле **uctx_drelease**. Если значение этого поля равно NULLF, то вызов операции не происходит.
- В режиме работы “Динамика” происходит объединение результатов, полученных на разных изображениях. В этом случае для связанных с результатами контекстов вызывается операция, указанная в поле **uctx_change**. Если значение этого поля равно NULLF, вызов операции не происходит.
- Поле **uctx_udata** может указывать на интересующие пользователя данные.

6.5 vodi_plate_info_spec

6.5.1 Имя

struct vodi_plate_info_spec — спецификация результата распознавания.

6.5.2 Синтаксис

```
#include <Bo/aorp/Vpwtypes.h>

struct vodi_plate_info_spec {
    size_t           pis_plate_variantc;
    vodi_plate_t     pis_plate_variantv[VodiL_VPW_VARIANT_MAX];

    vodi_rect_t      pis_outer_rect;
    vodi_rect_t      pis_exact_rect;
    vodi_quad64f_t   pis_exact_quad;

    _Bool            pis_inversed;
    vodi_motiondir_t pis_direction;
    double           pis_angle;
    double           pis_speed;
    size_t           pis_index;
    unsigned         pis_flags;
    _Bool            pis_move_in;
    size_t           pis_status;

    vodi_ucontext_t  pis_uctx;
    vodi_image_t     pis_image;

    bo_utime_t       pis_bftime;
    bo_utime_t       pis_ctime;
    bo_utime_t       pis_ltime;

    vodi_size_t      pis_imgsz;
    u_int32_t        pis_seqnum;
};
```

6.5.3 Описание

Таблица 16: Параметры struct vodi_plate_info_spec

Параметр	**Описание**
pis_plate_variantc	Количество вариантов номерной пластины.
pis_plate_variantv	Массив вариантов номерных пластин.
pis_outer_rect	Прямоугольная область с номером (“рамка номера”), заданная относительно изображения. Тип данного поля (vodi_rect_t) представляет прямоугольники, у которых стороны ортогональны координатным осям. Данное поле заполняется при базовом поиске номера, поэтому его значение может не иметь высокую точность (например, рамка может захватывать фары автомобиля, радиатор).
pis_exact_rect	Прямоугольная область с номером (“рамка номера”), заданная относительно изображения. Тип данного поля (vodi_rect_t) представляет прямоугольники, у которых стороны ортогональны координатным осям. В отличие от **pis_outer_rect**, данный параметр вычислен более точным образом и указывает на область минимальной площади номера, захватывающую его символы. Данное поле присутствует в структуре **vodi_plate_info_spec** для обратной совместимости, а для описания рамки номера используется поле **pis_exact_quad**.
pis_exact_quad	Прямоугольная область с номером (“рамка номера”), заданная относительно изображения. В отличие от **pis_outer_rect** и **pis_exact_rect**, которые имеют тип vodi_rect_t, данная прямоугольная область имеет большую точность и описывает наклон номерной пластины (тип vodi_quad64f_t).
pis_inversed	Признак инверсного номера.
pis_direction	Направление движения номерной пластины.
pis_angle	Угол наклона номерной пластины.
pis_speed	Скорость движения номерной пластины. Зарезервировано.
pis_index	Идентификатор номерной пластины.
pis_flags	Доступны флажки:
	VodiF_RESULT_DUP
	VodiF_RESULT_LOST
	VodiF_RESULT_INVALID
pis_move_in	Зарезервировано.
pis_status	Зарезервировано.
pis_uctx	Контекст пользователя.
pis_image	Изображение номерной пластины.
pis_bftime	Временная метка лучшего кадра с распознанным номером.
pis_ctime	Временная метка кадра, на котором данный номер был распознан впервые.

Параметр	Описание
pis_ltime	Временная метка кадра, на котором данный номер был распознан в последний раз.
pis_imgsz	Зарезервировано.
pis_seqnum	Зарезервировано.

6.6 vodi_plate

6.6.1 Имя

struct vodi_plate — номерная пластина.

6.6.2 Синтаксис

```
#include <Bo/aorp/Vpwtypes.h>

struct vodi_plate {
    wchar_t             pv_plate_string[VodiL_VPW_SYMBOL_MAX + 1];
    wchar_t             pv_plate_type[VodiL_VPW_SYMBOL_MAX + 5];
    u_int32_t           pv_plate_id;
    u_int32_t           pv_country_id;
    u_int32_t           pv_subdivision_id;
    float               pv_validity;
    float               pv_validity_coeff;

    float               pv_min_symb_validity;
    float               pv_max_geom_deviation;
    float               pv_max_symb_validity;
    float               pv_avg_symb_validity;

    vodi_plateid_t      pv_tmpl_id;

    vodi_color_t        pv_background;
    vodi_color_t        pv_sym_color;
    size_t              pv_symbolc;
    vodi_plate_symbol_t pv_symbolv[VodiL_VPW_SYMBOL_MAX];
    void_rect_t         pv_exact_rect;
};
```

6.6.3 Описание

Таблица 17: Параметры struct vodi_plate

Параметр	Описание
pv_plate_string	Юникод-строка номерной пластины.
pv_plate_type	Тип номера, задаваемый строкой:
	z Маленькая буква
	Z Большая буква
	x Цифра
	| Пробел между символами

Параметр	Описание
	A Буква или цифра
pv_plate_id	Идентификатор подходящего шаблона.
pv_country_id	Код страны (согласно ISO 3166-1 numeric).
pv_subdivision_id	Зарезервировано.
pv_tmpl_id	Идентификатор шаблона номерной пластины (тип — vodi_plateid_t), который включает в себя номер шаблона, код страны и код административного региона (если есть) согласно ISO 3166. Данное поле предназначено для использования вместо полей pv_plate_id, pv_country_id, pv_subdivision_id, которые в будущем будут считаться устаревшими. Функции для работы со структурами vodi_plateid_t описаны в разделе описание функций для работы с шаблонами.
pv_validity	Числовая оценка качества распознавания (по собственным критериям AutoSDK).
pv_validity_coeff	Зарезервировано.
pv_min_symb_validity	Зарезервировано.
pv_avg_symb_validity	Зарезервировано.
pv_max_geom_deviation	Зарезервировано.
pv_avg_geom_deviation	Зарезервировано.
pv_background	Цвет фона номерной пластины.
pv_sym_color	Цвет символов на номерной пластине.
pv_symbolc	Количество символов на номерной пластине.
pv_symbolv	Массив символов.
pv_exact_rect	Прямоугольная область с номером, заданная относительно изображения (область минимальной площади, которая ограничивает символы номерного знака).

6.7 aorp_error

6.7.1 Имя

struct aorp_error — описание ошибки.

6.7.2 Синтаксис

```
#include <Bo/aorp/Types.h>

struct aorp_error {
    char const   *func;
    char const   *file;
    long          line;
    bo_status_t   status;
    syserrcode_t  syserr;
    unsigned      msgidx;
    unsigned      bufsz;
    char         *msg;
};
```

6.7.3 Описание

Таблица 18: Параметры struct aorp_error

Параметр	Описание
func	Имя функции, при выполнении которой возникла ошибка. Опциональное поле, обычно заполняется только в отладочных версиях программ.
file	Путь к файлу, в котором реализована функция-триггер ошибки. Опциональное поле, обычно заполняется только в отладочных версиях программ.
line	Номер строки в файле **file**, в котором возникла ошибка. Опциональное поле. Имеет смысл, если **file** не равен NULL.
status	Содержит код ошибки (см. <Bo/Errors.h>) с условной областью, в которой эта ошибка возникла.
syserr	Код ошибки, принятый в данной операционной системе. Если ситуация не вынуждает использовать коды ОС, используются коды ошибок POSIX.
msgidx	Порядковый номер сообщения (см. <Bo/SMessages.h>).
bufsz	Размер буфера (в байтах), на который указывает поле **msg**.
msg	Указатель на строку, которая описывает ошибку.

7 ОПИСАНИЕ ФУНКЦИЙ ДЛЯ РАБОТЫ С ШАБЛОНАМИ

Соответствие старых и новых идентификаторов шаблонов номерных пластин[1] каждой страны записано в файле **doc/templates/templates_map.txt**[2].
Форма записи: ID_страны[.ID_региона].ID_шаблона:прежний_ID_шаблона
Также реализовано вспомогательное консольное приложение **lsvpwi**, которое печатает информацию обо всех или указанных шаблонах (см. Утилита lsvpwi).

7.1 vodiiso3166children

7.1.1 Имя

VodiISO3166children — получить список административных регионов указанного государства.

7.1.2 Синтаксис

```
#include <Vodi/ISO3166lib.h>

struct vodi_iso3166_elem const **
__attribute__((nonnull(1)))
VodiISO3166children(
    struct vodi_iso3166_elem const *anElem
    );
```

7.1.3 Описание

Функция **VodiISO3166children** принимает указатель на структуру vodi_iso3166_elem, которая описывает определенное государство (согласно стандарту ISO 3166-1).

7.1.4 Возвращаемые значения

В случае успеха функция **VodiISO3166children** возвращает массив указателей на структуры vodi_iso3166_elem, которые описывают административные регионы (согласно ISO 3166-2) указанного в аргументе государства. Массив указателей на структуры заканчивается NULL-указателем.

[1] новые идентификаторы реализованы в версии **AutoSDK** 2.5.1
[2] файл добавлен в версии **AutoSDK** 2.5.8

7.2 lpvlibshowplateid

7.2.1 Имя

LpvlibShowPlateid — преобразовать структуру-идентификатор шаблона номерной пластины в строку.

7.2.2 Синтаксис

```
#include <Vodi/services/Lpvlib.h>

char *
__attribute__((nonnull(1)))
LpvlibShowPlateid(
    char aResultPtr[/* LpvlibK_PLATEID_CHAR_MAX */],
    vodi_plateid_t aPlateid,
    aorp_opflags_t Flags
    );
```

7.2.3 Описание

Функция **LpvlibShowPlateid** принимает следующие параметры:

- **aResultPtr** — буфер, выделенный для результирующей строки.
- **aPlateid** — структура vodi_plateid_t, строковое представление которой необходимо получить.
- **Flags** — формат результирующей строки; если передается флаг **LpvlibF_HUMAN_READABLE**, код страны и региона будут иметь буквенное представление, если 0 — цифровое.

7.2.4 Возвращаемые значения

В случае успеха функция **LpvlibShowPlateid** возвращает указатель на строку, описывающую определенный в аргументе шаблон. В ином случае функция вернет NULL.

7.3 lpvlibreadplateid

7.3.1 Имя

LpvlibReadPlateid — преобразовать строку в структуру-идентификатор шаблона номерной пластины.

7.3.2 Синтаксис

```
#include <Vodi/services/Lpvlib.h>

vodi_plateid_t
__attribute__((nonnull(1)))
LpvlibReadPlateid(
    char const *aSrc,
    struct aorp_error *anErrPtr
    );
```

7.3.3 Описание

Функция **LpvlibReadPlateid** принимает указатель на строку, которая содержит описание шаблона номерной пластины, а также указатель на структуру aorp_error.

7.3.4 Возвращаемые значения

В случае успеха функция **LpvlibReadPlateid** возвращает структуру vodi_plateid_t. Она будет содержать данные, прочитанные из переданной аргументом строки (при условии, что эта строка синтаксически корректна). В ином случае функция запишет значение ошибки в структуру, переданную аргументом **anErrPtr** и вернет **VodiK_INVAL_PLATEID**.

7.4 lpvlibmodulesof

7.4.1 Имя

LpvlibModulesOf — получить список vpwi-модулей, которые содержат заданное параметром aSet множество шаблонов; определить, какой минимальный набор модулей достаточно загрузить, когда необходимо работать с неким множеством шаблонов.

7.4.2 Синтаксис

```
#include <Vodi/services/Lpvlib.h>

bo_seqlist_(char *) *
__attribute__((nonnull(2)))
LpvlibModulesOf(
    aorp_object_t aSelf /* = NULL */,
    bo_seqlist_(char *) *aResultPtr,
    char const *aSet /* = NULL */,
    struct aorp_error *anErrPtr
    );
```

7.4.3 Описание

Функция **LpvlibModulesOf** принимает следующие параметры:

- **aSet** — строковое представление множества шаблонов.
- **aResultPtr** — указатель на результирующий список.
- **anErrPtr** — указатель на структуру описания ошибки.

Пример функции загрузки необходимых модулей согласно множества шаблонов:

```
#include <Bo/fundis/Seqlist.h>
#include <Vodi/services/Lpvlib.h>

static bo_status_t
load_modules(
    char const *aTemplates,
    _Bool Force,
    struct aorp_error *anErrPtr
    )
{
    bo_status_t status;
    bo_seqlist_t mdlst;

    /* loading the vpwi module which contains the implementation of LpvlibModulesOf */
```

```
    status = AorpMldLoad("vpwi", NULL, 0, false, anErrPtr);
    if (BoS_FAILURE(status)) {
        return (status);
        /* NOTREACHED */
    }

    status = BoS_ERR;
    BO_SEQLIST_Init(&mdlst);
    if (NULL != LpvlibModulesOf(NULL, &mdlst, aTemplates, anErrPtr)) {
        BO_SEQLIST_FOREACH_(char *, &mdlst)
            status = AorpMldLoad(*_$1, NULL, 0, false, anErrPtr);
            if (BoS_FAILURE(status) && !Force)
                break;
        FOREACH_END
    }
    LpvlibStrlstDestroy(&mdlst);

    return (Force ? BoS_NORMAL : status);
}
```

7.4.4 Возвращаемые значения

В случае успеха функция возвращает указатель aResultPtr. В ином случае функция запишет значение ошибки в структуру, переданную аргументом anErrPtr и вернет NULL.

8 ОПИСАНИЕ ГРАММАТИКИ ДЛЯ ОБОЗНАЧЕНИЯ МНОЖЕСТВА ШАБЛОНОВ НОМЕРНЫХ ПЛАСТИН

Приведенная ниже грамматика используется следующей функциональностью:

- функциями семейства Lpvlib;
- командами функции VodiprincControl:
  - VodiCTL_GETVPWI
  - VodiCTL_ADDVPWI
  - VodiCTL_SETVPWI
  - VodiCTL_DELVPWI
- утилитой vpwfetch (как аргумент настройки -e, --templates);
- утилитой lsvpwi (в составе пользовательских команд);
- утилитой lsvpwc (в составе пользовательских команд).

```
1   E    : T1 \ ... \ Tn          (1 <= n, left associative) ;
2   T    : F1 ... Fn              (0 <= n) ;
3   F    : * | pse | ( E ) | ! F ;
4
5   pse  : oid | oid . cs | cs ;
6   oid  : code1 . ... . coden    (1 <= n) ;
7   cs   : [crs] | [^crs] ;
8   crs  : cr1 , ... , crn        (0 <= n) ;
9   cr   : code | code - code | code - inf ;
10  code : INT | ISO3166-name ;
```

- E (*expression*) — стартовый символ.
- \ — оператор разности множества шаблонов.
- T — подмножества, неявно объединенные пробелом.
- * — универсум.
- *pse* — plates' ID set (набор идентификаторов шаблонов).
- *oid* — objects' ID set (набор числовых кодов стран).
- *cs* — code set (набор кодов).
- *crs* — code range set (набор диапазонов кодов).
- *cr* — code range (диапазон кодов).

Для задания множества шаблонов могут использоваться как строковые, так и числовые коды стран согласно ISO 3166 (например, "ua.3 "804.3").
Примеры:

- **784.3** — выбрать все доступные шаблоны номеров[1], принадлежащие эмирату Дубай (ОАЭ).
- **784.3.**[**1-2**] либо **ae.du.**[**1,2**] — выбрать первые два шаблона номеров, принадлежащих эмирату Дубай (ОАЭ).
- **ae.du.**[**3,1**] либо **784.3.1 784.3.3** — выбрать шаблоны эмирата Дубай (ОАЭ) с порядковыми номерами 1 и 3.
- **ua** — выбрать все шаблоны номеров Украины.
- **784 804** либо **ua ae** — выбрать все шаблоны номеров Украины и ОАЭ.
- [**1-19**] [**31-inf**] — выбрать шаблоны номеров всех стран, коды которых не лежат в диапазоне 20-30.
- **784.**[**1-2**] **784.3.**[**11-inf**] **784.**[**4-inf**] либо **ae\ae.du.**[**1-10**] — выбрать все шаблоны ОАЭ, кроме шаблонов эмирата Дубай с порядковыми номерами в диапазоне 1-10.
- **!zm** либо [**1-893**] [**895-inf**] — выбрать все доступные шаблоны, кроме тех, что принадлежат Замбии.
- **!(zm zw)** либо [**1-715**] [**717-893**] [**895-inf**] — выбрать все доступные шаблоны, кроме тех, что принадлежат Замбии и Зимбабве.

[1] поддерживаемые **AutoSDK**