



Руководство по интеграции (HTTP API, IIDK, ActiveX, HTTP
Сервер, Axxon Next)

Обновлено 03.04.2024

Содержание

1	Руководство по интеграции ПК Интеллект. Введение	17
2	Интеграция аппаратно-программных модулей.....	18
3	Элемент управления ActiveX CamMonitor.ocx.....	19
4	HTTP API ПК Интеллект	20
5	HTTP-сервер ПК Интеллект	22
6	Интеграция с Axxon Next	23
7	Настройка RabbitMQ	24
8	Заключение	25
9	ПРИЛОЖЕНИЕ 1. Описание структуры ddi-файла.....	26
10	ПРИЛОЖЕНИЕ 2. Объявление классов NissObjectDLLExt и CoreInterface.....	27
11	Руководство по интеграции ПК Интеллект. Введение	28
12	Интеграция аппаратно-программных модулей.....	29
12.1	INTELLECT INTEGRATION DEVELOPER KIT (IIDK).....	29
12.1.1	Общие сведения об IIDK	29
12.1.1.1	Назначение IIDK	29
12.1.1.2	Требования к разработчику	29
12.1.1.3	Состав IIDK	29
12.1.2	Подключение к ПК Интеллект	30
12.1.2.1	Параметры подключения	30
12.1.2.2	Объект Интерфейс IIDK.....	30
12.1.2.3	Настройка пересылки событий через объект Интерфейс IIDK	31
12.1.2.4	Особенности интеграции банкоматов. Объект Банкомат	32
12.1.3	Функции IIDK	33
12.1.3.1	Connect	33
12.1.3.2	SendMsg	34
12.1.3.3	Disconnect.....	34
12.1.3.4	Другие функции.....	35
12.1.3.4.1	Connect3	35
12.1.3.4.2	SendReactToCore	35
12.1.3.4.3	IsConnected	36
12.1.3.4.4	Connect4	36

12.1.3.4.5	SendData4	37
12.1.3.4.6	SendFile	37
12.1.3.4.7	GetMsg	37
12.1.3.4.8	SetPingTime	37
12.1.4	Синтаксис отправляемых сообщений.....	38
12.1.4.1	Синтаксис сообщений.....	38
12.1.4.2	Синтаксис сообщений (900 порт).....	39
12.1.4.3	Использование классов Event и React	40
12.1.5	Примеры управления объектами системы.....	40
12.1.5.1	Добавление, изменение и удаление объектов системы	40
12.1.5.1.1	Добавление пользователя в отдел	41
12.1.5.1.2	Добавление и удаление устройства видеоввода.....	41
12.1.5.2	Особенности работы с системой в многопользовательском режиме.....	41
12.1.5.3	Определение компьютера, на котором был выгружен ПК Интеллект (через 1030 порт).....	42
12.1.5.4	Вывод видеокамеры на монитор.....	42
12.1.5.5	Получение параметров объекта (через 1030 порт). GET_CONFIG	42
12.1.5.6	Получение информации о состоянии объекта. GET_STATE и GET_LIST	43
12.1.5.7	Вывод информационного сообщения. SET_STATE	44
12.1.5.8	Работа с живым и архивным видео	44
12.1.5.8.1	Получение живого видео	44
12.1.5.8.2	Получение архивного видео	44
12.1.5.8.3	Получение списка временных интервалов.....	44
12.1.5.9	Управление телеметрией	45
12.1.5.10	Операции со слоем карты.....	45
12.1.5.11	Получение информации об очередях ядра командой GET_QUEUE_INFO.....	46
12.1.5.12	Проигрывание аудиоархива за период. Команда START_PLAY_TIME	46
12.1.5.13	Воспроизведение тонового набора. Команды START_TONE и STOP_TONE.....	47
12.1.5.14	Получение статистики по видеопотоку. STATISTIC.....	47
12.2	Интеграция аппаратно-программных модулей с ПК Интеллект	47
12.2.1	Общие сведения об интеграции аппаратно-программных модулей.....	47
12.2.2	Редактирование DBI-файла.....	48
12.2.2.1	Добавление объектов в intellect.dbi	48
12.2.2.2	Использование утилиты ddi.exe для работы с DBI-файлами	51
12.2.3	Редактирование DDI-файла.....	52
12.2.3.1	Добавление в intellect.ddi информации об объекте	53

12.2.3.2	Использование утилиты ddi.exe для работы с DDI-файлами	54
12.2.4	Дополнительные возможности утилиты ddi.exe	56
12.2.5	Разработка MDL-файла	58
12.2.6	Разработка RUN-файла	64
12.2.7	Создание и настройка интегрированных объектов (модулей) в ПК Интеллект	65
13	Элемент управления ActiveX CamMonitor.ocx.....	66
13.1	Общее описание ActiveX-компонента CamMonitor.ocx	66
13.1.1	Общее описание CamMonitor.ocx	66
13.1.2	Требования к разработчику	66
13.2	Установка CamMonitor.ocx	66
13.3	Параметры CamMonitor.ocx.....	67
13.3.1	CamMenuOptions	67
13.3.2	CamMenuProcessingOptions	68
13.3.3	CamButtonsOptions	68
13.3.4	MainPanelOptions.....	69
13.3.5	KeysOptions	69
13.3.6	OverlayMode	69
13.3.7	Пример использования параметров.....	70
13.4	Методы CamMonitor.ocx	70
13.4.1	Connect	70
13.4.2	ShowCam	71
13.4.3	DoReactMonitor	71
13.4.4	RemoveAllCams	72
13.4.5	IsConnected	72
13.4.6	GetCurlp.....	72
13.4.7	SendRawMessage	72
13.4.8	Disconnect.....	72
13.4.9	SetCallBackOptions	72
13.4.10	SetParam	73
13.5	События CamMonitor.ocx	73
14	HTTP API ПК Интеллект	74
14.1	Общие сведения о HTTP API	74
14.1.1	Программная реализация	74
14.1.2	Возможности.....	74

14.1.3	Особенности и ограничения.....	74
14.1.4	Авторизация	75
14.1.5	Формат ответа по умолчанию	75
14.1.6	Кросс-доменные запросы (CORS)	75
14.1.7	Использование HTTPS.....	75
14.2	Версия продукта	75
14.2.1	Общий формат запроса:.....	75
14.2.2	Пример запроса:	75
14.2.3	Пример ответа:.....	75
14.3	Авторизация в ПК Интеллект по token ключу	76
14.3.1	Общий формат запроса:.....	76
14.3.2	Параметры запроса:	76
14.3.3	Пример запроса:	76
14.3.4	Пример ответа:.....	76
14.3.5	Параметры ответа:.....	77
14.4	Карта	77
14.4.1	Получение списка карт	77
14.4.1.1	Общий формат запроса:.....	77
14.4.1.2	Пример запроса:	77
14.4.1.3	Пример ответа:.....	77
14.4.1.4	Параметры ответа.....	78
14.4.2	Информация об одной карте.....	80
14.4.2.1	Общий формат запроса:.....	80
14.4.2.2	Параметры запроса:	80
14.4.2.3	Пример запроса:	80
14.4.2.4	Пример ответа:.....	80
14.4.2.5	Параметры ответа:.....	81
14.4.3	Список слоёв для выбранной карты	81
14.4.3.1	Общий формат запроса:.....	81
14.4.3.2	Параметры запроса	81
14.4.3.3	Пример запроса:	81
14.4.3.4	Пример ответа:.....	81
14.4.3.5	Параметры ответа.....	82
14.4.4	Информация о конкретном слое	83
14.4.4.1	Общий формат запроса:.....	83

14.4.4.2	Параметры запроса:	84
14.4.4.3	Пример запроса:	84
14.4.4.4	Пример ответа:.....	84
14.4.4.5	Параметры ответа:.....	84
14.4.5	Фоновый рисунок слоя.....	85
14.4.5.1	Общий формат запроса.....	85
14.4.5.2	Параметры запроса	85
14.4.5.3	Пример запроса.....	85
14.4.5.4	Пример ответа.....	85
14.4.5.5	Ошибки выполнения запроса.....	85
14.4.6	Список точек на слое	85
14.4.6.1	Общий формат запроса:.....	85
14.4.6.2	Параметры запроса:	85
14.4.6.3	Пример запроса:	85
14.4.6.4	Пример ответа:.....	86
14.4.6.5	Параметры ответа:.....	86
14.4.7	Информация об отдельной точке на слое	87
14.4.7.1	Общий формат запроса:.....	87
14.4.7.2	Параметры запроса:	87
14.4.7.3	Пример запроса:	87
14.4.7.4	Пример ответа:.....	87
14.4.7.5	Параметры ответа:.....	88
14.5	Классы объектов	88
14.5.1	Список классов объектов, которые существуют на сервере.....	88
14.5.1.1	Общий формат запроса:.....	88
14.5.1.2	Пример запроса:	88
14.5.1.3	Пример ответа:.....	88
14.5.1.4	Параметры ответа:.....	89
14.5.2	Отдельный класс объектов.....	89
14.5.2.1	Общий формат запроса:.....	89
14.5.2.2	Параметры запроса:	89
14.5.2.3	Пример запроса:	89
14.5.2.4	Пример ответа:.....	89
14.5.2.5	Параметры ответа:.....	89
14.5.3	Список состояний для определённого класса объектов.....	89

14.5.3.1	Общий формат запроса:.....	89
14.5.3.2	Параметры запроса:	89
14.5.3.3	Пример запроса:	90
14.5.3.4	Пример ответа:.....	90
14.5.3.5	Параметры ответа:.....	90
14.5.4	Информация о конкретном состоянии.....	90
14.5.4.1	Общий формат запроса:.....	90
14.5.4.2	Параметры запроса:	90
14.5.4.3	Пример запроса:	90
14.5.4.4	Пример ответа:.....	90
14.5.4.5	Параметры ответа:.....	91
14.5.5	Получение иконки для определённого состояния	91
14.5.5.1	Общий формат запроса:.....	91
14.5.5.2	Параметры запроса:	91
14.5.5.3	Пример запроса:	91
14.5.5.4	Пример ответа:.....	91
14.5.6	Список событий для определенного класса объектов	91
14.5.6.1	Общий формат запроса:.....	91
14.5.6.2	Параметры запроса:	91
14.5.6.3	Пример запроса:	91
14.5.6.4	Пример ответа:.....	91
14.5.6.5	Параметры ответа:.....	92
14.6	Объекты	92
14.6.1	Получение списка всех объектов сервера.....	92
14.6.1.1	Общий формат запроса:.....	92
14.6.1.2	Параметры запроса:	92
14.6.1.3	Пример запроса:	92
14.6.1.4	Пример ответа:.....	93
14.6.1.5	Параметры ответа:.....	97
14.6.2	Информация об отдельном объекте	98
14.6.2.1	Общий формат запроса:.....	98
14.6.2.2	Параметры запроса:	98
14.6.2.3	Пример запроса:	99
14.6.2.4	Пример ответа:.....	99
14.6.2.5	Параметры ответа:.....	99

14.6.3	Состояние отдельного объекта	99
14.6.3.1	Общий формат запроса:.....	99
14.6.3.2	Параметры запроса:	99
14.6.3.3	Пример запроса:	99
14.6.3.4	Пример ответа:.....	99
14.6.3.5	Параметры ответа:.....	100
14.6.4	Список доступных действий с объектом, находящимся в определённом состоянии.....	101
14.6.4.1	Общий формат запроса:.....	101
14.6.4.2	Параметры запроса:	101
14.6.4.3	Пример запроса:	101
14.6.4.4	Пример ответа:.....	101
14.6.4.5	JSON	102
14.6.4.6	Параметры ответа:.....	102
14.6.5	Получение списка всех областей и регионов	103
14.6.5.1	Общий формат запроса:.....	103
14.6.5.2	Пример запроса:	103
14.6.5.3	Пример ответа:.....	103
14.6.5.4	Параметры ответа:.....	104
14.7	Получение событий	104
14.7.1	Общий формат запроса:.....	104
14.7.2	Пример запроса:	105
14.7.3	Пример ответа:.....	105
14.7.4	Параметры ответа:.....	105
14.7.5	Получение событий видеоподсистемы блоками	106
14.7.5.1	Общий формат запроса:.....	106
14.7.5.2	Параметры запроса:	106
14.7.5.3	Пример запроса:	106
14.7.5.4	Пример ответа:.....	106
14.7.5.5	Коды возврата:	107
14.8	Отсылка команд на сервер	108
14.8.1	Общий формат запроса:.....	108
14.8.2	Параметры запроса:	108
14.8.3	Пример запроса:	108
14.9	Макрокоманды	108
14.9.1	Получение списка макрокоманд.....	108

14.9.1.1	Общий формат запроса:.....	108
14.9.1.2	Пример запроса:.....	108
14.9.1.3	Пример ответа:.....	108
14.9.1.4	Параметры ответа:.....	109
14.9.2	Получение параметров макрокоманд.....	109
14.9.2.1	Общий формат запроса:.....	109
14.9.2.2	Параметры запроса:.....	109
14.9.2.3	Пример запроса:.....	109
14.9.2.4	Пример ответа:.....	109
14.9.2.5	Параметры ответа:.....	109
14.9.3	Запрос на выполнение макрокоманды на сервере.....	109
14.9.3.1	Общий формат запроса:.....	109
14.9.3.2	Параметры запроса:.....	110
14.9.3.3	Пример запроса:.....	110
14.10	Видео.....	110
14.10.1	Запрос миниатюр (скриншотов).....	110
14.10.1.1	Общий формат запроса:.....	110
14.10.1.1.1	1-й способ.....	110
14.10.1.2	Параметры запроса:.....	110
14.10.1.2.1	2-й способ.....	110
14.10.1.3	Параметры запроса:.....	111
14.10.1.4	Пример запроса:.....	111
14.10.1.4.1	1-й способ.....	111
14.10.1.4.2	2-й способ.....	111
14.10.1.5	Пример ответа:.....	111
14.10.2	Запрос конфигурации.....	111
14.10.2.1	Общий формат запроса:.....	111
14.10.2.2	Параметры запроса:.....	112
14.10.2.3	Пример запроса:.....	112
14.10.2.4	Пример ответа:.....	112
14.10.2.5	Параметры ответа:.....	112
14.10.3	Запрос видео.....	113
14.10.3.1	Общий формат запроса:.....	113
14.10.3.2	Параметры запроса:.....	113
14.10.3.3	Пример запроса:.....	114

14.10.3.4	Пример ответа:.....	114
14.10.4	Curl запрос видео. Формат основного потока	114
14.10.4.1	Общий формат запроса:.....	114
14.10.4.2	Параметры запроса:	114
14.10.4.3	Пример запроса:	115
14.10.4.4	Пример ответа:.....	115
14.10.4.5	Параметры ответа:.....	116
14.10.4.6	Пример ответа:.....	116
14.10.4.7	Параметры ответа:.....	116
14.10.4.8	Пример ответа:.....	116
14.10.4.9	Параметры ответа:.....	117
14.10.5	Управление камерой	117
14.10.5.1	Общий формат запроса:.....	117
14.10.5.2	Параметры запроса:	117
14.10.5.3	Пример запроса:	118
14.10.6	Получение авторизованной ссылки на камеру по token ключу.....	118
14.10.6.1	Общий формат запроса для получения токена:	118
14.10.6.2	Параметры запроса:	118
14.10.6.3	Пример запроса:	119
14.10.6.4	Пример ответа:.....	119
14.10.6.5	Параметры ответа:.....	119
14.10.6.6	Общий формат запроса по полученному токenu через Веб-сервер 2.0:	119
14.10.6.7	Параметры запроса:	119
14.10.6.8	Пример запроса:.....	119
14.10.6.9	Общий формат запроса по полученному токenu через Web сервер 1:.....	120
14.10.6.10	Параметры запроса:	120
14.10.6.11	Пример запроса:.....	120
14.11	Управление телеметрией.....	120
14.11.1	Общий формат запроса:.....	120
14.11.2	Параметры запроса:	120
14.11.3	Пример запроса:	121
14.12	Работа с архивом	121
14.12.1	Получение списка записей (1-й способ).....	121
14.12.1.1	Параметры запроса:	122
14.12.1.2	Пример запроса:	122

14.12.1.3	Пример ответа:.....	122
14.12.2	Получение списка записей (2-й способ).....	123
14.12.2.1	Общий формат запроса:.....	123
14.12.2.2	Параметры запроса:	123
14.12.2.3	Пример запроса:	123
14.12.2.4	Пример ответа:.....	123
14.12.3	Получение видео из архива с использованием команды "arc.play"	125
14.12.3.1	Общий формат запроса:.....	125
14.12.3.2	Параметры запроса:	125
14.12.3.3	Пример запроса:	126
14.12.3.4	Пример ответа:.....	126
14.12.4	Получение одного кадра из архива с использованием команды "arc.frame"	126
14.12.4.1	Общий формат запроса (1-й способ):.....	126
14.12.4.2	Параметры запроса:	126
14.12.4.3	Пример запроса:	127
14.12.4.4	Общий формат запроса (2-й способ):.....	127
14.12.4.5	Параметры запроса:	127
14.12.4.6	Пример запроса:	127
14.12.4.7	Пример ответа:.....	128
14.12.5	Экспорт архива	128
14.12.5.1	Создание задания на экспорт архива.....	128
14.12.5.1.1	Общий формат запроса:.....	128
14.12.5.1.2	Параметры запроса:	128
14.12.5.1.3	Пример запроса:	129
14.12.5.1.4	Пример ответа:.....	129
14.12.5.1.5	Параметры ответа:.....	129
14.12.5.2	Получение статуса экспорта	129
14.12.5.2.1	Общий формат запроса:.....	129
14.12.5.2.2	Параметры запроса:	130
14.12.5.2.3	Пример запроса:	130
14.12.5.2.4	Пример ответа:.....	130
14.12.5.2.5	Параметры ответа:.....	130
14.12.5.3	Удаление файла архива.....	130
14.12.5.3.1	Общий формат запроса:.....	130
14.12.5.3.2	Параметры запроса:	131

14.12.5.3.3	Пример запроса:	131
14.12.5.3.4	Пример ответа:.....	131
14.12.5.3.5	Параметры ответа:.....	131
14.13	Звук	131
14.13.1	Получение живого звука	131
14.13.1.1	Общий формат запроса:.....	131
14.13.1.2	Параметры запроса:	131
14.13.1.3	Пример запроса:	132
14.13.1.4	Пример ответа:.....	132
14.13.2	Остановка получения живого звука.....	133
14.13.2.1	Общий формат запроса:.....	133
14.13.2.2	Параметры запроса:	133
14.13.2.3	Пример запроса:	133
14.13.2.4	Пример ответа:.....	133
14.13.3	Проигрывание звука из архива.....	133
14.13.3.1	Общий формат запроса:.....	133
14.13.3.2	Параметры запроса:	134
14.13.3.3	Пример запроса:	134
14.13.3.4	Пример ответа:.....	134
14.13.4	Отправка живого звука.....	134
14.13.4.1	Общий формат запроса:.....	134
14.13.4.2	Параметры запроса:	134
14.13.4.3	Пример запроса:	135
14.13.4.4	Пример ответа:.....	135
14.13.4.5	Параметр ответа:.....	135
14.14	Пользователи	135
14.14.1	Получение списка пользователей.....	135
14.14.1.1	Общий формат запроса:.....	135
14.14.1.2	Пример запроса:	135
14.14.1.3	Пример ответа:.....	135
14.15	Команды, используемые для интеграции ЕЦХД	141
14.15.1	Получение информации об устройстве.....	142
14.15.1.1	Общий формат запроса:.....	142
14.15.1.2	Пример запроса:	142
14.15.1.3	Пример ответа:.....	142

14.15.1.4	Параметры ответа:.....	142
14.15.2	Получение списка идентификаторов камер.....	142
14.15.2.1	Общий формат запроса:.....	142
14.15.2.2	Пример запроса:.....	142
14.15.2.3	Пример ответа:.....	142
14.15.2.4	Параметры ответа:.....	143
14.15.3	Диапазоны доступных архивных записей.....	143
14.15.3.1	Общий формат запроса:.....	143
14.15.3.2	Параметры запроса:.....	143
14.15.3.3	Пример запроса:.....	143
14.15.3.4	Пример ответа:.....	143
14.15.4	Работа с видеопотоками	144
14.15.4.1	Запрос GetLiveUrl.....	144
14.15.4.1.1	Общий формат запроса:.....	144
14.15.4.1.2	Параметры запроса:.....	144
14.15.4.1.3	Пример запроса:.....	144
14.15.4.1.4	Пример ответа:.....	144
14.15.4.2	Запрос GetArchiveURL	144
14.15.4.2.1	Общий формат запроса:.....	144
14.15.4.2.2	Параметры запроса:.....	145
14.15.4.2.3	Пример запроса:.....	145
14.15.4.2.4	Пример ответа:.....	145
14.15.5	Выгрузка архивов	145
14.15.5.1	Общий формат запроса:.....	145
14.15.5.2	Параметры запроса:.....	145
14.15.5.3	Пример запроса:.....	145
14.15.5.4	Пример ответа:.....	145
14.15.6	Экспорт архивных записей ЕЦХД	146
14.15.7	Управление функциями средства видеонаблюдения.....	146
14.15.7.1	Общий формат запроса:.....	146
14.15.7.2	Параметры запроса	146
14.15.7.3	Пример запроса:.....	148
14.15.7.4	Пример ответа:.....	148
14.15.7.5	Параметры ответа:.....	148
14.15.8	Пример использования команд ЕЦХД при работе с NAT	148

14.15.8.1	Общий формат запроса:.....	149
14.15.8.2	Параметры запроса:	149
14.15.8.3	Пример запроса:	149
14.15.8.4	Пример ответа:.....	149
14.15.8.5	Параметры ответа:.....	150
14.16	Отправка реакций, событий и xml-данных в ПК Интеллект через HTTP API	150
14.16.1	Отправка реакций в ПК Интеллект через HttpListener.....	150
14.16.1.1	Общий формат запроса:.....	150
14.16.1.2	Параметры запроса:	150
14.16.1.3	Пример запроса:	150
14.16.2	Отправка событий об отсканированном штрихкоде	151
14.16.2.1	Общий формат запроса:.....	151
14.16.2.2	Параметры запроса:	151
14.16.2.3	Пример запроса:	151
14.16.2.4	Пример ответа:.....	151
14.16.3	Отправка реакций и событий в ПК Интеллект по HTTP-запросу	151
14.16.3.1	Общий формат запроса:.....	151
14.16.3.2	Параметры запроса:	152
14.16.3.3	Примеры запроса:	152
14.16.4	Пример отправки команд HTTP API из утилиты curl	153
14.16.4.1	Пример запроса:	153
14.16.4.2	Пример ответа:.....	153
14.16.5	Отправка xml-файла	153
14.16.5.1	Общий формат запроса:.....	153
14.16.5.2	Параметры запроса:	154
14.16.5.3	Пример запроса:	154
14.16.5.4	Пример ответа:.....	154
14.16.5.5	Параметры ответа:.....	154
14.17	Настройка интеграции с Техносерв.....	154
14.17.1	Заполнение конфигурационного файла ApiBgConfig.xml	155
14.17.2	Примеры команд для работы с интеграцией Техносерв	157
14.17.2.1	Запрос на получение списка камер.....	157
14.17.2.1.1	Общий формат запроса:.....	157
14.17.2.1.2	Пример запроса:	157
14.17.2.1.3	Пример ответа:.....	158

14.17.2.2	Запрос на получение списка подписок.....	158
14.17.2.2.1	Общий формат:.....	158
14.17.2.2.2	Пример запроса:.....	158
14.17.2.2.3	Пример ответа:.....	158
14.17.2.3	Запрос на создание подписки.....	158
14.17.2.3.1	Общий формат:.....	158
14.17.2.3.2	Пример запроса:.....	159
14.17.2.3.3	Пример ответа:.....	159
14.17.2.4	Запрос на удаление одной подписки.....	159
14.17.2.4.1	Общий формат:.....	159
14.17.2.4.2	Параметры запроса:.....	159
14.17.2.4.3	Пример запроса:.....	159
14.17.2.5	Запрос на удаление всех подписок.....	159
14.17.2.5.1	Общий формат:.....	159
14.17.2.5.2	Пример запроса:.....	159
14.18	Обращение к API вертикальных решений через HTTP API ПК Интеллект.....	159
14.18.1	Общий формат запроса.....	160
14.18.2	Параметры запроса.....	160
14.18.3	Примеры запросов.....	160
14.18.4	Примеры ответов.....	160
15	HTTP-сервер ПК Интеллект.....	161
15.1	Общие сведения о HTTP Сервере.....	161
15.2	Настройка объекта HTTP Сервер.....	161
15.3	Запросы к HTTP Серверу.....	162
15.3.1	Запрос идентификатора последнего события.....	162
15.3.2	Запрос событий с учетом фильтра.....	162
15.3.3	Запрос списка всех созданных считывателей.....	163
16	Интеграция с Axxon Next.....	165
16.1	Интеграция с Axxon Next по протоколу Onvif.....	165
16.1.1	Настройка объекта Устройство видеоввода для интеграции с Axxon Next по Onvif.....	165
16.1.1.1	С помощью Мастера создания IP-устройств.....	165
16.1.1.2	Создание объекта в Оборудовании.....	168
17	Настройка RabbitMQ.....	169
17.1	Поставщик ПК Интеллект и сторонний приемник.....	169

17.2	Приемник ядра ПК Интеллект	170
18	Заключение	171
19	ПРИЛОЖЕНИЕ 1. Описание структуры ddi-файла.....	172
20	ПРИЛОЖЕНИЕ 2. Объявление классов NissObjectDLLExt и CoreInterface.....	174
20.1	CoreInterface.....	174
20.2	NissObjectDLLExt.....	175

1 Руководство по интеграции ПК Интеллект. Введение

2 Интеграция аппаратно-программных модулей

- INTELLECT INTEGRATION DEVELOPER KIT (IIDK)
 - Общие сведения об IIDK
 - Назначение IIDK
 - Требования к разработчику
 - Состав IIDK
 - Подключение к ПК Интеллект
 - Параметры подключения
 - Объект Интерфейс IIDK
 - Настройка пересылки событий через объект Интерфейс IIDK
 - Особенности интеграции банкоматов. Объект Банкомат
 - Функции IIDK
 - Connect
 - SendMsg
 - Disconnect
 - Другие функции
 - Синтаксис отправляемых сообщений
 - Синтаксис сообщений
 - Синтаксис сообщений (900 порт)
 - Использование классов Event и React
 - Примеры управления объектами системы
 - Добавление, изменение и удаление объектов системы
 - Особенности работы с системой в многопользовательском режиме
 - Определение компьютера, на котором был выгружен ПК Интеллект (через 1030 порт)
 - Вывод видеокамеры на монитор
 - Получение параметров объекта (через 1030 порт). GET_CONFIG
 - Получение информации о состоянии объекта. GET_STATE и GET_LIST
 - Вывод информационного сообщения. SET_STATE
 - Работа с живым и архивным видео
 - Управление телеметрией
 - Операции со слоем карты
 - Получение информации об очередях ядра командой GET_QUEUE_INFO
 - Проигрывание аудиоархива за период. Команда START_PLAY_TIME
 - Воспроизведение тонового набора. Команды START_TONE и STOP_TONE
 - Получение статистики по видеопотоку. STATISTIC
- Интеграция аппаратно-программных модулей с ПК Интеллект
 - Общие сведения об интеграции аппаратно-программных модулей
 - Редактирование DBI-файла
 - Добавление объектов в intellect.dbi
 - Использование утилиты ddi.exe для работы с DBI-файлами
 - Редактирование DDI-файла
 - Добавление в intellect.ddi информации об объекте
 - Использование утилиты ddi.exe для работы с DDI-файлами
 - Дополнительные возможности утилиты ddi.exe
 - Разработка MDL-файла
 - Разработка RUN-файла
 - Создание и настройка интегрированных объектов (модулей) в ПК Интеллект

3 Элемент управления ActiveX CamMonitor.ocx

- [Общее описание ActiveX-компонента CamMonitor.ocx](#)
- [Установка CamMonitor.ocx](#)
- [Параметры CamMonitor.ocx](#)
- [Методы CamMonitor.ocx](#)
- [События CamMonitor.ocx](#)

4 HTTP API ПК Интеллект

- Общие сведения о HTTP API
- Версия продукта
- Авторизация в ПК Интеллект по token ключу
- Карта
 - Получение списка карт
 - Информация об одной карте
 - Список слоёв для выбранной карты
 - Информация о конкретном слое
 - Фоновый рисунок слоя
 - Список точек на слое
 - Информация об отдельной точке на слое
- Классы объектов
 - Список классов объектов, которые существуют на сервере
 - Отдельный класс объектов
 - Список состояний для определённого класса объектов
 - Информация о конкретном состоянии
 - Получение иконки для определённого состояния
 - Список событий для определённого класса объектов
- Объекты
 - Получение списка всех объектов сервера
 - Информация об отдельном объекте
 - Состояние отдельного объекта
 - Список доступных действий с объектом, находящимся в определённом состоянии
 - Получение списка всех областей и регионов
- Получение событий
 - Получение событий видеоподсистемы блоками
- Отсылка команд на сервер
- Макрокоманды
 - Получение списка макрокоманд
 - Получение параметров макрокоманд
 - Запрос на выполнение макрокоманды на сервере
- Видео
 - Запрос миниатюр (скриншотов)
 - Запрос конфигурации
 - Запрос видео
 - Curl запрос видео. Формат основного потока
 - Управление камерой
 - Получение авторизованной ссылки на камеру по token ключу
- Управление телеметрией
- Работа с архивом
 - Получение списка записей (1-й способ)
 - Получение списка записей (2-й способ)
 - Получение видео из архива с использованием команды "arc.play"
 - Получение одного кадра из архива с использованием команды "arc.frame"
 - Экспорт архива
 - Создание задания на экспорт архива
 - Получение статуса экспорта
 - Удаление файла архива
- Звук
 - Получение живого звука
 - Остановка получения живого звука
 - Проигрывание звука из архива
 - Отправка живого звука
- Пользователи
 - Получение списка пользователей
- Команды, используемые для интеграции ЕЦХД

- Получение информации об устройстве
- Получение списка идентификаторов камер
- Диапазоны доступных архивных записей
- Работа с видеопотоками
 - Запрос GetLiveUrl
 - Запрос GetArchiveURL
- Выгрузка архивов
- Экспорт архивных записей ЕЦХД
- Управление функциями средства видеонаблюдения
- Пример использования команд ЕЦХД при работе с NAT
- Отправка реакций, событий и xml-данных в ПК Интеллект через HTTP API
 - Отправка реакций в ПК Интеллект через HttpListener
 - Отправка событий об отсканированном штрихкоде
 - Отправка реакций и событий в ПК Интеллект по HTTP-запросу
 - Пример отправки команд HTTP API из утилиты curl
 - Отправка xml-файла
- Настройка интеграции с Техносерв
 - Заполнение конфигурационного файла ApiBgConfig.xml
 - Примеры команд для работы с интеграцией Техносерв
 - Запрос на получение списка камер
 - Запрос на получение списка подписок
 - Запрос на создание подписки
 - Запрос на удаление одной подписки
 - Запрос на удаление всех подписок
- Обращение к API вертикальных решений через HTTP API ПК Интеллект

5 HTTP-сервер ПК Интеллект

- [Общие сведения о HTTP Сервере](#)
- [Настройка объекта HTTP Сервер](#)
- [Запросы к HTTP Серверу](#)

6 Интеграция с Axxon Next

- [Интеграция с Axxon Next по протоколу Onvif](#)
 - [Настройка объекта Устройство видеоввода для интеграции с Axxon Next по Onvif](#)

7 Настройка RabbitMQ

8 Заключение

9 ПРИЛОЖЕНИЕ 1. Описание структуры ddi-файла

10 ПРИЛОЖЕНИЕ 2. Объявление классов NissObjectDLLExt и CoreInterface

Скачать модуль DEMO



Примечание.

Данный модуль используется в качестве примера работы с IIDK – см. [Интеграция аппаратно-программных модулей](#).

11 Руководство по интеграции ПК Интеллект. Введение

В руководстве приведена информация о том, как обеспечить взаимодействие ПК *Интеллект* с внешними системами. ПК *Интеллект* предоставляет следующие решения:

1. **Интерфейс IIDK.** Используется для внедрения в систему функциональных модулей, обеспечивающих решение следующих задач:
 - a. Добавление нового охранного оборудования в систему.
 - b. Реализация новых сервисных функций (управление охранным оборудованием).Этапы интеграции модулей рассмотрены на примере демонстрационного модуля *DEMO*, исходные файлы которого приложены к документации.
Скачать модуль *DEMO* можно на странице [Руководство по интеграции ПК Интеллект](#).
2. **Элемент управления ActiveX CamMonitor.ocs.** Данный компонент является полным аналогом интерфейсного объекта **Монитор видеонаблюдения**. Он позволяет управлять камерами, просматривать архив и т.д.
3. **HTTP API.** Данный программный интерфейс позволяет отправлять команды и получать данные от ПК *Интеллект* при помощи HTTP-запросов.
4. **RabbitMQ.** Позволяет отправлять и получать сообщения в ПК *Интеллект* с помощью брокера сообщений RabbitMQ.
5. **Интеграция с Axxon Next.** Позволяет получать данные от видеокамер *Axxon Next*.

Примечание

Интеграция с ЕЦХД описана в [Приложение 5. Настройка ПК Интеллект для использования интеграции с ЕЦХД](#).

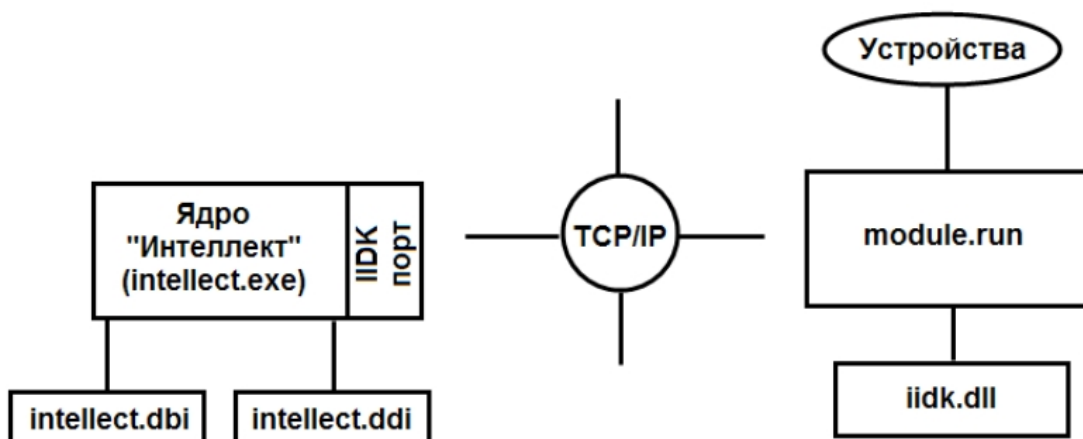
12 Интеграция аппаратно-программных модулей

12.1 INTELLECT INTEGRATION DEVELOPER KIT (IIDK)

12.1.1 Общие сведения об IIDK

12.1.1.1 Назначение IIDK

Возможность расширить систему заложена в архитектуру программного комплекса *Интеллект*, предусматривающую межзадачное взаимодействие ядра системы с функциональными модулями (смежными информационными системами) через коммуникационную среду TCP/IP. Схема взаимодействия ядра ПК *Интеллект* с внешним программным обеспечением (функциональным модулем) приведена на рисунке.



Взаимодействие ядра системы с внешним программным обеспечением выполняется посредством обмена сообщениями в коммуникационной среде, реализованного с помощью *IIDK*.

Intellect Integration Developer Kit (IIDK) представляет собой комплект средств разработки, используемый для интеграции охранного оборудования сторонних производителей с ПК *Интеллект*. Данный инструмент позволяет быстро и эффективно расширять систему, добавляя функциональные модули, поддерживающие новое оборудование или новые сервисные функции.

12.1.1.2 Требования к разработчику

Для использования *IIDK* требуется:

1. знание языка программирования C/C++ ;
2. знание основ программирования в Win32;
3. наличие среды разработки (*Microsoft Visual C++*, *C++ Builder*, *DELPHI* и др.), поддерживающей работу с dll-файлами.

Примечание.

Создавая lib-файл в C++ Builder 5 при помощи утилиты `implib.exe`, необходимо указать ключ '- a'.

12.1.1.3 Состав IIDK

IIDK включает в себя следующие средства разработки:

1. *iidk.ocx* – элемент управления ActiveX. Данный файл при установке ПК *Интеллект* помещается в папку `Windows\System32` и регистрируется в операционной системе.
2. *ddi.exe* – программа для просмотра и редактирования DDI- и DBI- файлов. Располагается в папке <Директория установки ПК *Интеллект*>\Tools.

12.1.2 Подключение к ПК Интеллект

12.1.2.1 Параметры подключения

Взаимодействие ядра ПК *Интеллект* с функциональными модулями (смежными информационными системами) осуществляется со следующими параметрами подключения:

1. Номер порта.
 - a. Для видеоподсистемы – порт 900.
 - b. Для объекта **Интерфейс IIDK** – порт 1030.
 - c. Для объекта **Банкомат** – порт 1009.

Примечание.

Для подключения банкоматов можно также использовать не порт 1009 (АТМ), а порт 1030 (IIDK), в таком случае объект **Банкомат** будет отображаться в дереве оборудования со значком красного креста. При этом в дереве оборудования должен быть создан объект **Интерфейс IIDK**.

2. IP-адрес компьютера, на котором функционирует ядро ПК *Интеллект*.
3. ID – идентификатор объекта подключения.

Внимание!

Для подключения к видеоподсистеме (порт 900) id должен быть больше 1 и не должен совпадать с id созданных в системе объектов **Интерфейс IIDK**. Для подключения к объекту **Интерфейс IIDK** (порт 1030) id равен идентификационному номеру объекта, заданному в диалоговом окне настройки ПК *Интеллект*.

Примечание.

Если требуется подключиться к серверу (объекту **Интерфейс IIDK**) с удаленного компьютера, не обязательно устанавливать ПК *Интеллект* на удаленный компьютер, но необходимо добавить этот компьютер в конфигурацию ПК *Интеллект* на сервере (на вкладке **Оборудование** диалогового окна **Настройка системы**), и именно на базе созданного объекта **Компьютер** создать объект **Интерфейс IIDK**. В таком случае в параметре IP функции Connect следует указывать адрес сервера, а в параметре ID идентификатор указанного объекта Интерфейс IIDK. Следует учитывать, что объект **Компьютер**, соответствующий удаленному компьютеру, будет помечен в дереве объектов красным крестом.

Примечание.

Если создан mdl-файл (см. раздел [Разработка MDL-файла](#)), для подключения к ядру ПК *Интеллект* объект **Интерфейс IIDK** в системе не создается. В качестве идентификатора подключения передается пустая строка, то есть id равен “”.

12.1.2.2 Объект Интерфейс IIDK

Объект **Интерфейс IIDK** позволяет управлять всеми элементами системы. Объект **Интерфейс IIDK** создается на базе объекта **Компьютер** в дереве объектов ПК *Интеллект*.

Примечание

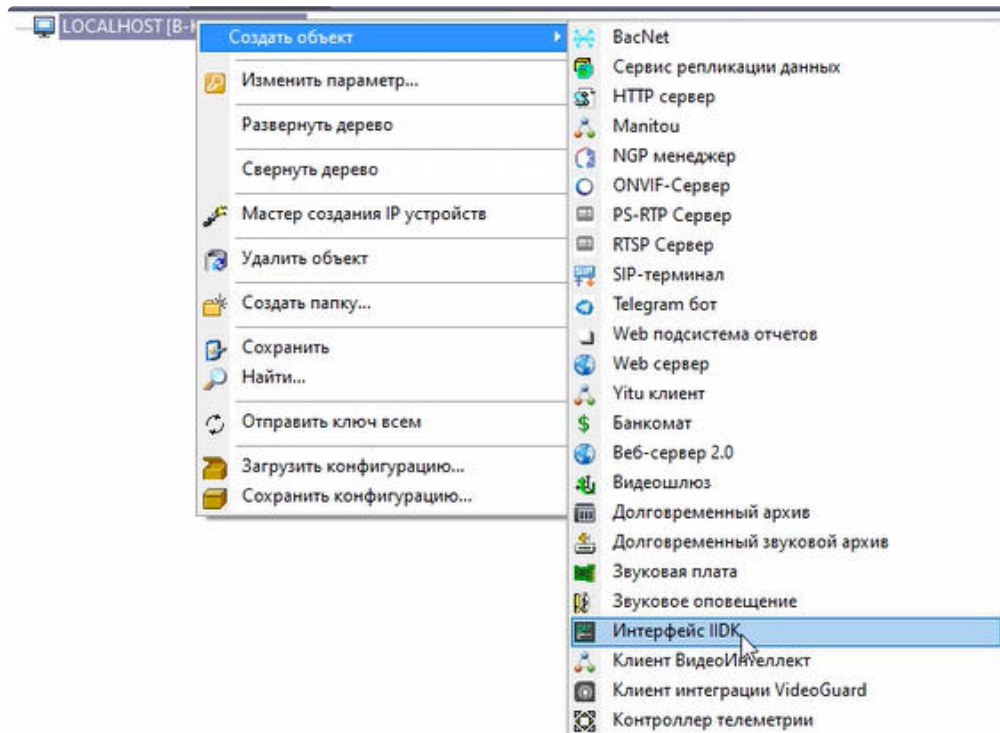
Для использования объекта **Интерфейс IIDK** данная функциональная возможность должна быть разрешена в ключе активации.

Примечание

Если ПК *Интеллект* запущен в демонстрационном режиме, объект **Интерфейс IIDK** будет активирован после подключения функционального модуля к ядру системы (см. раздел [Connect](#)).

⚠ Внимание!

Идентификатор объекта **Интерфейс IIDK** не должен совпадать с идентификаторами созданных в системе объектов **Монитор**.



В случае использования объекта **Интерфейс IIDK** панели настройки для интегрируемых функциональных модулей (смежного программного обеспечения) не создаются.

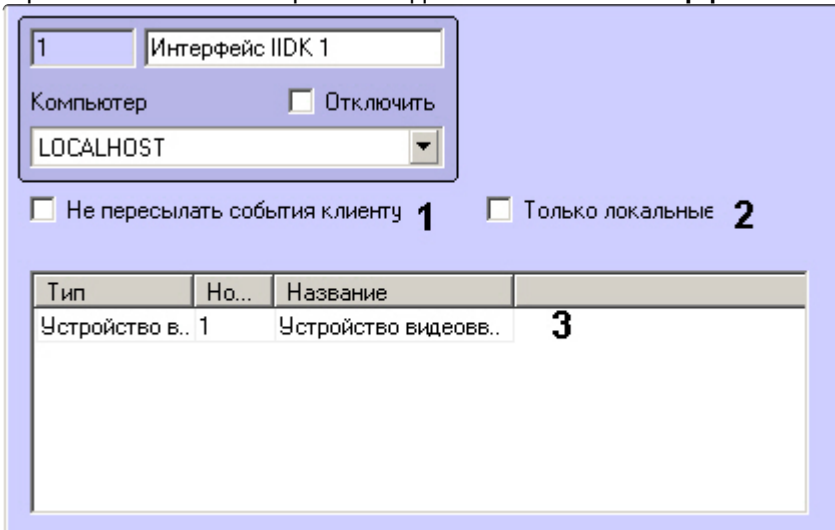
При использовании распределенной архитектуры ПК *Интеллект* объект **Интерфейс IIDK** должен быть создан на компьютере, содержащем программное ядро, к которому выполняется подключение. В случае, если подключение выполняется к компьютеру, на котором установлено *Рабочее место мониторинга*, то в параметрах подключения требуется указывать ip-адрес *Сервера* или *Рабочего места администратора*.

12.1.2.3 Настройка пересылки событий через объект Интерфейс IIDK

Объект **Интерфейс IIDK** позволяет настроить фильтрацию событий, передаваемых подключаемым клиентским приложениям.

Настройка фильтрации осуществляется следующим образом:

1. Перейти на панель настройки созданного объекта **Интерфейс IIDK**.



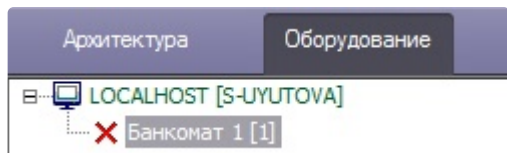
Тип	Но...	Название
Устройство в..	1	Устройство видеовв..

2. Если не требуется передавать никакие сообщения клиентским приложениям, от которых не поступают запросы к ядру, установить флажок **Не пересылать события клиенту** (1).
3. Если требуется передавать клиентским приложениям только события от объектов, созданных на базе того же объекта **Компьютер**, что и объект **Интерфейс IIDK**, установить флажок **Только локальные** (2).
4. В таблице (3) задать список объектов, события от которых следует передавать подключающимся клиентским приложениям. Для фильтрации событий от ядра необходимо указать тип CORE.

Настройка фильтрации событий завершена.

12.1.2.4 Особенности интеграции банкоматов. Объект Банкомат

Для передачи событий от ПО банкомата в ядро ПК *Интеллект* может быть использован объект **Банкомат**. Данный объект создается на базе объекта **Компьютер** на вкладке **Оборудование** диалогового окна **Настройка системы** вместо объекта **Интерфейс IIDK**.



Объект **Банкомат** позволяет отображать события банкомата ("Вставлена карта", "Отдана карта" и др.) в протоколе событий. Данные события также могут быть использованы для наложения титров на видео, настройки реакций и т.д.

Примечание.

Для событий, связанных с картой клиента, имеется возможность передавать маскированный номер банковской карты в параметре param0.

Список доступных для использования событий объекта **Банкомат** можно узнать при помощи утилиты ddi.exe. Работа с данной утилитой описана в документе [Руководство Администратора](#), раздел [Утилита редактирования шаблонов баз данных и файла внешних настроек ddi.exe](#).

Способ подключения и синтаксис сообщений для объекта **Банкомат** такие же, как для объекта **Интерфейс IIDK**, однако для отправки сообщений используется порт 1009 (см. также [Параметры подключения](#) и [Синтаксис сообщений \(900 порт\)](#)).

12.1.3 Функции IIDK

12.1.3.1 Connect

Для взаимодействия функционального модуля с ПК *Интеллект* необходимо выполнить подключение к ядру системы с помощью следующей функции:

```
BOOL Connect (LPCTSTR ip, LPCTSTR port, LPCTSTR id, void (_stdcall *func)(LPCTSTR msg))
```

Описание параметров функции Connect приведено в таблице.

Параметр	Описание	Пример
LPCTSTR ip	IP-адрес компьютера с ядром системы	<pre>CString port = "900"; CString ip = "127.0.0.1"; CString id = "2"; BOOL IsConnect = Connect(ip, port, id, myfunc); if (!IsConnect) { // не удалось подключиться AfxMessageBox("Error"); }</pre>
LPCTSTR port	порт TCP/IP, через который происходит подключение	
LPCTSTR id	идентификатор подключения для видео	
_stdcall *func)(LPCTSTR msg)	Callback-функция, принимающая сообщения от ПК <i>Интеллект</i>	

Функция возвращает TRUE, если подключение выполнено, иначе – FALSE.

Все сообщения, приходящие от ядра системы, принимает Callback-функция.

Пример объявления Callback-функции:

```
void _stdcall myfunc(LPCTSTR str)
{
    printf("\r\nReceived:%s\r\n\r\n",str);
}
```

Примечание.

Void _stdcall myfunc вызывается в отдельном потоке (не в контексте основного потока приложения).

Разбор получаемых сообщений устанавливается разработчиком в соответствии с требованиями интеграции.

12.1.3.2 SendMsg

Для передачи сообщения ядру системы используется функция:

```
BOOL SendMsg (LPCTSTR id, LPCTSTR msg)
```

Описание параметров функции SendMsg приведено в таблице.

Параметр	Описание	Пример
LPCTSTR id	идентификатор подключения, указанный при вызове функции Connect	<pre>CString port = "9000"; CString ip = "127.0.0.1"; CString id = "2"; BOOL IsConnect = Connect(ip, port, id, myfunc); if (!IsConnect) { // не удалось подключиться AfxMessageBox("Error"); Return; } SendMsg(id, "CAM 1 REC"); // поставить камеру 1 на запись Disconnect (id);</pre>
LPCTSTR msg	текст сообщения	

Если сообщение отправлено, функция возвращает TRUE, иначе – FALSE.

12.1.3.3 Disconnect

Каждое созданное соединение должно быть разорвано с помощью функции **Disconnect**:

```
void Disconnect (LPCTSTR id)
```

где **LPCTSTR id** – идентификатор подключения, указанный при вызове функции **Connect**.

Если разрыв соединения осуществляется со стороны ПК *Интеллект*, то Callback-функция принимает значение **DISCONNECTED**.

Примечание.

Пример использования функции **Disconnect** приведен в разделе [SendMsg](#).

12.1.3.4 Другие функции

На странице:
<ul style="list-style-type: none"> • Connect3 • SendReactToCore • IsConnected • Connect4 • SendData4 • SendFile • GetMsg • SetPingTime

Ниже перечислены дополнительные функции, объявленные в заголовочном файле iidk.h. Из них не рекомендуются к использованию функции Connect4, SendData4, SendFile, GetMsg. Они созданы для внутреннего пользования. Функция Connect2 не используется.

12.1.3.4.1 Connect3

<pre>BOOL Connect3(LPCTSTR ip, LPCTSTR port, LPCTSTR id, iidk_callback_func* lpfunc, DWORD user_param, int async_connect, DWORD connect_attempts)</pre>	
Параметр	Описание
ip	IP-адрес Сервера ПК <i>Интеллект</i> ,
port	Порт TCP/IP, через которое происходит подключение
id	Идентификатор подключения slave, для видео
lpfunc	Callback-функция, принимающая сообщения от ПК <i>Интеллект</i>
user_param	Дополнительный параметр, который будет приходить в Callback-функцию, чтобы разделить слайвы, если функция одна на всех
async_connect	0 - синхронный режим подключения, функция возвращает TRUE, если подключение выполнено -1 - асинхронный режим подключения, функция всегда возвращает FALSE, если подключение выполнено, то генерируется событие CONNECTED Любое другое значение - сначала используется синхронный режим, в случае неудачи асинхронный
connect_attempts	Количество попыток подключения

12.1.3.4.2 SendReactToCore

Функция предназначена для отправки реакции в указанное ядро.

```
BOOL SendReactToCore(LPCTSTR id, LPCTSTR msg)
```

Параметр	Описание
id	Идентификатор подключения ядра
msg	Отправляемое сообщение. Формат сообщения аналогичен SendMsg

12.1.3.4.3 IsConnected

IsConnected возвращает TRUE, если в данный момент клиент подключен к серверу.

```
BOOL IsConnected();
```

12.1.3.4.4 Connect4

```
BOOL Connect4(LPCTSTR ip, LPCTSTR port, LPCTSTR id, iidk_callback_func* lpfunc,
              iidk_frame_callback_func* lpframe_func, iidk_user_data_func*
              iidk_user_data_func,
              DWORD user_param, int async_connect, DWORD connect_attempts);
```

Параметр	Описание
ip	IP-адрес Сервера ПК <i>Интеллект</i>
port	Порт TCP/IP, через которое происходит подключение
id	Идентификатор подключения ядра, для видео
lpfunc	Callback-функция, принимающая сообщения от ПК <i>Интеллект</i>
lpframe_func	Callback-функция, принимающая видеокadres
iidk_user_data_func	Callback-функция для данных, посылаемых при помощи функции SendData4
user_param	Дополнительный параметр, который будет приходить в Callback-функцию, чтобы разделить ядра, если Callback-функция одна на все ядра.
async_connect	0 - синхронный режим подключения, функция возвращает TRUE, если подключение выполнено -1 - асинхронный режим подключения, функция всегда возвращает FALSE. Если подключение выполнено, то генерируется событие CONNECTED Любое другое значение - сначала используется синхронный режим, в случае неудачи асинхронный режим
connect_attempts	Количество попыток подключения

12.1.3.4.5 SendData4

Данная функция используется для отправки CUserNetObject, ее назначение - отправка "сырых данных".

```
BOOL SendData4(LPCTSTR id, int nIdent, BYTE *pBuffer, DWORD dwSize);
```

Параметр	Описание
id	Идентификатор подключения ядра
nIdent	Уникальный идентификатор данных
pBuffer	Пересылаемые данные
dwSize	Размер массива данных

12.1.3.4.6 SendFile

Функция служит для пересылки файла.

```
BOOL SendFile(LPCTSTR id, LPCTSTR file_from, LPCTSTR file_to)
```

Параметр	Описание
id	Идентификатор подключения ядра
file_from	Адрес, по которому находится файл для пересылки
file_to	Адрес, по которому следует записать файл

12.1.3.4.7 GetMsg

Функция служит для выборки пришедших сообщений, которые помещаются в очередь, если Callback-функция не указана.

```
BOOL GetMsg(LPTSTR msg, DWORD& cb)
```

Параметр	Описание
msg	Получаемое сообщение
cb	Длина сообщения

12.1.3.4.8 SetPingTime

Функция включает и задает интервал отправки KeepAlive сообщений в ядро ПК *Интеллект*. Достаточно вызвать функцию один раз, например, после вызова CreateClient.

```
void SetPingTime(intptr_t clientId, unsigned int time);
```

Параметр	Описание
clientId	Идентификатор клиента
time	Интервал отправки KeepAlive сообщений в миллисекундах. Минимальное значение 5000; если будет задано меньшее значение (но не 0), то будет использовать значение 5000. Если задать 0, то отсылка KeepAlive сообщений прекращается

12.1.4 Синтаксис отправляемых сообщений

12.1.4.1 Синтаксис сообщений

Сообщения, отправляемые ядру, должны иметь следующий вид:

```
CORE||DO_REACT|source_type<ТИП ОБЪЕКТА>,source_id<ИДЕНТИФИКАТОР ОБЪЕКТА>,action<ДЕЙСТВИЕ>  
[ ,params<КОЛ-ВО ПАРАМЕТРОВ>,param0_name<ИМЯ ПАРАМЕТРА_0>,param0_val<ЗНАЧЕНИЕ ПАРАМЕТРА_0>]
```

Ниже приведен синтаксис сообщения, содержащего 2 параметра.

```
CORE||DO_REACT|source_type<ТИП ОБЪЕКТА>,source_id<ИДЕНТИФИКАТОР  
ОБЪЕКТА>,action<ДЕЙСТВИЕ>,params<2>,param0_name<ИМЯ ПАРАМЕТРА_0>,param0_val<ЗНАЧЕНИЕ  
ПАРАМЕТРА_0>,param1_name<ИМЯ ПАРАМЕТРА_1>,param1_val<ЗНАЧЕНИЕ ПАРАМЕТРА_1>
```

Описание параметров сообщения приведено в таблице.

Параметр	Описание
source_type<obj>	тип объекта (см. DDI-файл, секцию [OBJTYPE])
source_id<id>	идентификационный номер объекта, заданный при создании объекта в ПК <i>Интеллект</i> (см. дерево настроек в ПК <i>Интеллект</i>)
action<react>	действие (см. DDI-файл, секцию [REACT])
params<number>	число передаваемых параметров в десятичном формате
param0_name<str1>	имя параметра
param0_val<str2>	значение параметра

Примечание.

Для работы с DDI-файлами предпочтительно использовать программу ddi.exe (см. раздел [Использование утилиты ddi.exe для работы с DDI-файлами](#)).

Пример. Отправление сообщения с командой перевода телеметрии в предустановку 4.

```
CString msg=
```

```
"CORE||DO_REACT|
source_type<TELEMETRY>,source_id<1.1>,action<G0_PRESET>,params<2>,param0_name<preset>,param0_val<4>,param1_name<tel_prior>,param1_val<2>";

SendMsg(id,msg);
```

12.1.4.2 Синтаксис сообщений (900 порт)

Сообщения, отправленные на 900 порт, передаются видеоподсистеме напрямую, поэтому сообщения имеют другой синтаксис.

Сообщения, отправляемые видеоподсистеме, имеют следующий вид:

ТИП ОБЪЕКТА|ИДЕНТИФИКАТОР ОБЪЕКТА|ДЕЙСТВИЕ [|ПАРАМЕТР<ЗНАЧЕНИЕ>]

Ниже описан синтаксис сообщения для видеоподсистемы, содержащего n-ое количество параметров.

ТИП ОБЪЕКТА|ИДЕНТИФИКАТОР ОБЪЕКТА|ДЕЙСТВИЕ [|ПАРАМЕТР 1<ЗНАЧЕНИЕ>,ПАРАМЕТР 2<ЗНАЧЕНИЕ>,...,ПАРАМЕТР N<ЗНАЧЕНИЕ>]

Внимание!

Через 900 порт можно управлять только объектами типа GRABBER, CAM и MONITOR.

Описание параметров сообщения представлено в таблице:

Параметр	Описание
Тип объекта	Тип объекта (GRABBER, CAM или MONITOR)
Идентификатор объекта	Идентификационный номер объекта, заданный при создании объекта в ПК <i>Интеллект</i>
Действие	Действие (команда)
Параметр <Значение>	Имя параметра. В треугольных скобках задается значение параметра

Для проверки сети можно использовать параметр со значением <trash:N>, где N – длина параметра в байтах. В результате в сообщении будет сгенерирована текстовая строка заданной длины, содержащая цифры и латинские буквы. Имя у параметра может быть любое. Например, в результате добавления к сообщению

```
param1<trash:128>
```

в ответе будет сгенерировано

```
param1<88n9ESCAfYbFHyXzIoPpOF1GirSmLMJ1Ahh9oYZDIQou2RgWY0jXlaL2HhaX9ZY2oKfkQYLuPLg8bGNQK60umvQjVp1G7BEt1wFsw6FApzIZ3NbBUJNCKnP3350lgEwN>
```

Пример 1. Постановка камеры 1 на запись.

```
CString msg = "CAM|1|REC";

SendMsg (id,msg);
```

Пример 2. Запись видео со всех видеокамер на локальный диск «C:».

```
CString msg = "GRABBER|1|SET_DRIVES|drives<C:\>" ;
SendMsg(id,msg);
```

Примечание

Для выполнения команды **SET_DRIVES** необходимо указать идентификационный номер любой устройства видеоввода, созданной в системе.

Примечание

Команда **SET_DRIVES** не меняет настройки записи видеосигнала в архив, заданные в системе.

12.1.4.3 Использование классов Event и React

Для работы с сообщениями можно использовать прилагаемые классы: *Event* и *React*, объявленные в файле msg.h.

Сообщение, составленное без использования классов	Сообщение, составленное с помощью класса React
<pre>CString msg = "CORE DO_REACT source_type<TELEMETRY>,source_id<1.1>, action<GO_PRESET>,params<2>,param0_name<preset>,param0_val<4>, param1_name<tel_prior>,param1_val<2>"; SendMsg(id,msg);</pre>	<pre>React react("TELEMETRY","1.1","GO_PRESET"); react.SetParamInt("preset",4); react.SetParamInt("tel_prior",2); SendMsg(id,react.MsgToString().c_str());</pre>

Примечание.

Файлы msg.h и msg.cpp содержатся в папке Misc.

12.1.5 Примеры управления объектами системы

Внимание!

Команды и параметры объектов подробно описаны в документе [Руководство по программированию](#).

12.1.5.1 Добавление, изменение и удаление объектов системы

На странице:

- [Добавление пользователя в отдел](#)
- [Добавление и удаление устройства видеоввода](#)

Добавление, изменение и удаление объектов системы выполняется с помощью команд:

1. **CORE||CREATE_OBJECT** – для создания нового объекта.
2. **CORE||UPDATE_OBJECT** – для изменения существующего объекта или создания нового.
3. **CORE||DELETE_OBJECT** – для удаления объекта.

12.1.5.1.1 Добавление пользователя в отдел

Ниже приведено сообщение, в результате обработки которого в отдел будет добавлен пользователь с заданными параметрами:

```
CORE||CREATE_OBJECT|
objtype<PERSON>,objid<12341>,parent_id<1>,surname<Tim>,name<Kovac>,card<12362>,facility_code<0>
```

В ответ на эту команду IIDK вернет сообщение вида:

```
CORE||CREATE_OBJECT|card<1234>,objtype<PERSON>,guid_pk<{281A172C-62D2-EA11-A54B-B06EBF811A34}>,
facility_code<122>,surname<Tim>,module<iidk_client_test_x64.exe>,time<16:42:53>,parent_id<1>,fraction<797>,date<30-07-20>,
name<Kovac>,owner<QA-T49>,slave_id<QA-T49.11>,objid<12341>
```

Это необходимо для того, чтобы получить идентификатор созданного объекта в параметре objid<>.

12.1.5.1.2 Добавление и удаление устройства видеоввода

Добавление объекта выполняется с помощью команды **UPDATE_OBJECT**, если в системе отсутствует объект с указанными значениями для параметров **objtype** и **objid**.

```
CORE||UPDATE_OBJECT|objtype<GRABBER>,objid<12>,core_global<0>,parent_id<SLAVAXP>,name<Устройство видеоввода 1>,
params<5>,param0_name<format>,param0_val<NTSC>,param1_name<mode>,param1_val<1>,param2_name<chan>,param2_val<2>,
param3_name<type>,param3_val<FX 4>,param4_name<resolution>,param4_val<0>
```

Получив следующее сообщение, система изменит имя созданного объекта:

```
CORE||UPDATE_OBJECT|objtype<GRABBER>,objid<12>,core_global<0>,parent_id<SLAVAXP>,name<Устройство 2>
```

Для удаления объекта и всех его дочерних объектов используется команда **DELETE_OBJECT**:

```
CORE||DELETE_OBJECT|objtype<GRABBER>,objid<12>
```

12.1.5.2 Особенности работы с системой в многопользовательском режиме

На удаленном компьютере должен быть установлен (тип установки – **Клиент**) и запущен ПК *Интеллект*, для того чтобы обмениваться сообщениями с Сервером.

Если в ПК *Интеллект* созданы пользователи и настроены права доступа, то передаваемые сообщения, требующие ответа от ядра системы, должны содержать параметр **receiver_id<ID>**, где ID – это идентификационный номер объекта **Интерфейс IIDK** в системе.

```
CORE||GET_CONFIG|objtype<CAM>,objid<1>,receiver_id<1>
```

// Возвращает параметры объекта «Камера 1»

Чтобы получить идентификатор пользователя и его прав по его логину и паролю, используется функция CHECK_USER. Примеры использования:

```
CORE||CHECK_USER|password<1>,login<1>
```

```
CORE||CHECK_USER|pass_key<1373503546>,login<1> (crc32 из БД)
```

```
CORE||CHECK_USER|md5<bf03b1605e3c83978514f2a6546eef50> (md5 из БД)
```

Ответ:

```
ACTIVEX|1|USER_RIGHTS|rights_id<1>,user_id<1>
```

В случае, если указан неправильный пароль, ответ приходит с задержкой в 1 секунду.

12.1.5.3 Определение компьютера, на котором был выгружен ПК Интеллект (через 1030 порт)

В случае выгрузки ПК *Интеллект* в Callback-функцию придет сообщение, где параметру **action** присвоено значение **DISCONNECTED**:

```
ACTIVEX|12|EVENT|SOCKET<>,MMF<>,objaction<DISCONNECTED>,TRANSPORT_TYPE<MMF>,core_global<1>,
action<DISCONNECTED>, module<slave.exe>, objtype<SLAVE>,__slave_id<SLAVAXP.12>,
objid<SLAVAXP>,owner<SLAVAXP>,TRANSPORT_ID<1111>,time<12:41:16>,date<23-09-02>
```

Данное сообщение содержит имя компьютера, на котором был выгружен ПК *Интеллект*, дату и время, когда это действие произошло.

12.1.5.4 Вывод видеочамеры на монитор

Система удалит все чамеры с монитора и вызовет указанную видеочамеру, получив следующее сообщение:

```
CORE||DO_REACT|
source_type<MONITOR>,source_id<1>,action<REPLACE>,params<4>,param0_name<slave_id>,param0_val<SLAVA>,para
m1_name<cam>,param1_val<1>,param2_name<control>,param2_val<1>,param3_name<name>,param3_val<>
```

При подключении через 900 порт действие, описанное выше, выполняется с помощью сообщения:

```
MONITOR|1|REPLACE|slave_id<SLAVA>,cam<1>,control<1>
```

12.1.5.5 Получение параметров объекта (через 1030 порт). GET_CONFIG

Пример использования команды **GET_CONFIG** приведен ниже.

```
CORE||GET_CONFIG|objtype<CAM>,objid<1>
```

В возвращаемом сообщении будут содержаться все параметры указанного объекта:

```
ACTIVEX|12|OBJECT_CONFIG|
rec_priority<0>,mask0<>,decoder<0>,mask1<>,flags<>,mask2<>,compression<3>,sat_u<5>,mask3<>,proc_time<>,hot_re
c_period<>,mask4<>,telemetry_id<>,manual<1>,region_id<1.1>,contrast<5>,md_mode<0>,md_size<5>,audio_type<>,pr
e_rec_time<0>,config_id<>,bright<7>,alarm_rec<0>,audio_id<>,rec_time<>,hot_rec_time<2>,activity<>,mux<0>,parent_i
d<1>,objtype<CAM>,type<>,__slave_id<SLAVAXP.12>,objid<1>,name<Камера 1>,objname<Камера
1>,color<1>,priority<0>,md_contrast<5>
```

Примечание.

Если убрать параметр **objid**, то в Callback-функцию вернется конфигурация всех объектов заданного типа.

Пример. Получить данные пользователя по его идентификатору.

```
CORE||GET_CONFIG|objtype<PERSON>,objid<1>
```

В ответ придет сообщение, среди параметров которого находится требуемая информация, такая как имя пользователя, номер карты доступа и прочее:

```
ACTIVEX|1|OBJECT_CONFIG|
pnet3_sound<0>,galaxy_dual_focus<0>,auto_pass_type<>,galaxy_pin_change<0>,external_id<>,card_date<26.05.2017
10:57:06>,galaxy_tag_link<0>,rubeg8_zone_id<>,levels_times<>,expired<>,hid_escort_id<>,objtype<PERSON>,level2_id<>,galax
y_group_choice<0>,
who_level<>,hid_use_extended_access<0>,visit_purpose<>,card<1234>,email<>,galaxy_timer_schedule<0>,galaxy_menu_opti
on<0>,aiu_holiday<0>,
area_id<>,aiu_alarm<0>,objname<Пользователь 1>,surname<>,who_card<>,auto_brand<>,pnet3_alarm<0>,card_loss<0>,
facility_code<432>,galaxy_temp_code<0>,post<>,when_area_id_changed<>,drivers_licence<>,bolid_in_device<0>,pnet3_acs_
counter<0>,
temp_levels_times<>,temp_card<>,pnet3_no_entry<0>,location<>,temp_level_id<>,patronymic<>,teleph_work<>,department<
>,galaxy_keypad<0>,
_TRANSPORT_ID<>,finished_at<>,aiu_ksd_type<>,all_param<>,galaxy_template<0>,tabnum<>,parent_id<1>,pur<>,galaxy_dure
```

```
ss<0>,pnet3_no_exit<0>,
galaxy_dual<0>,hid_pin_exempt<0>,pnet3_counter<0>,whence<>,schedule_id<>,hid_enable_pin_commands<0>,galaxy_dual_a
ccess<0>,
pnet3_block<0>,passport<>,person<>,galaxy_menu_choice<0>,flags<0>,auto_number<>,phone<>,pin<>,rubeg8_AccessToBCPTi
meZoneNumber<>,
begin_temp_level<>,pnet3_master<0>,aiu_mark<0>,end_temp_level<>,visit_birthplace<>,galaxy_menu_level<>,visit_document
<>,pnet3_guard<0>,
pnet3_black<0>,aiu_kso_type<>,is_apb<0>,name<Пользователь 1>,pnet3_temp<0>,started_at<>,level_id<>,marker<>,
bolid_user_type<>,owner_person_id<>,card_type<>,is_active_temp_level<0>,begin<>,hid_line_tag<>,guid<{5B358685-E041-
E711-BCC6-DA0AE28E0C17}>,
pnet3_guest<0>,is_locked<0>,objid<1>,marketing_info<>,comment<>,aiu_vpu_arm<0>,visit_reg<>,bolid_in_pku<0>,pnet3_2car
ds_mask<0>
```

Примечание.

Данные пользователя также можно получить напрямую через запрос к базе данных ПК *Интеллект* из таблицы OBJ_PERSON. В этом случае можно использовать для выбора требуемого пользователя также номер карты. Подробнее о работе с базой данных ПК *Интеллект* см. [Руководство Администратора](#), раздел [Приложение 4. Администрирование базы данных программного комплекса Интеллект](#).

12.1.5.6 Получение информации о состоянии объекта. GET_STATE и GET_LIST

Для получения информации о состоянии объекта используется команда **GET_STATE**:

CORE||GET_STATE|objtype<CAM>,objid<1>

В результате возвратится строка:

ACTIVEV|12|OBJECT_STATE|objtype<CAM>,__slave_id<SLAVAXP.12>,objid<1>,state<DISARM_DETACHED>

Состояние указанного объекта будет представлено значением параметра **state** – одно из состояний, указанных в DDI-файле для выбранного объекта.

При подключении через 900 порт запрос состояний объектов выполняется с использованием команды **GET_LIST**:

CAM||GET_LIST

Примечание.

Независимо от того, указан идентификационный номер объекта или нет, команда возвратит состояния всех объектов заданного типа.

Возвращаемые сообщения имеют вид:

```
CAM|1|SETUP|rec_priority<0>,is_armed<0>,is_recorded<0>,bt<0>,slave_id<SLAVAXP>,compression<3>,sat_u<5>,
proc_time<0>,hot_rec_period<0>,manual<1>,telemetry_id<>,is_detached<1>,contrast<5>,md_size<5>,md_mode<0>,
is_alarmed<0>,audio_type<>,pre_rec_time<0>,bright<7>,audio_id<>,rec_time<0>,alarm_rec<0>,hot_rec_time<2>,
mux<0>,parent_id<1>,__slave_id<SLAVAXP>,priority<0>,mask<>,color<1>,md_contrast<5>,is_ring<1>,fs_error<0>
```

Состояния в сообщении представлены следующим образом: **is_state<val>**, где **state** – имя состояния объекта (см. DDI-файл); **val** – принимает значение 1, если объект находится в соответствующем состоянии, иначе – 0.

Примечание.

Параметр **is_ring<>** говорит о том, ведет ли камера запись в архив по кольцу. Параметр **fs_error** принимает значение 1 при наличии проблем с записью (например, не удастся удалить старую папку при записи по кольцу).

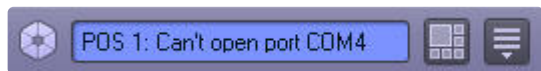
12.1.5.7 Вывод информационного сообщения. SET_STATE

Для вывода информационного сообщения на дисплей главной панели управления ПК *Интеллект* используется команда SET_STATE:

CORE|SET_STATE|name<POS 1>,value<Can't open port COM4>

Результат обработки сообщения системой представлен на рисунке.

Отображение сообщения на дисплее главной панели управления ПК *Интеллект*:



Удаление информационного сообщения с дисплея выполняется следующим образом:

CORE|SET_STATE|name<POS 1>,value<>

12.1.5.8 Работа с живым и архивным видео

12.1.5.8.1 Получение живого видео

Для получения живого видео с Камеры 1 следует отправить на порт 900 сообщение:

CAM|1|START_VIDEO|compress<1>

Здесь compress<> – степень компрессии, от 0 до 5. В ответ на это сообщение начнут приходить кадры видео. Пример программной обработки поступающих кадров можно найти в демо-комплекте, доступном для скачивания на странице [Руководство по интеграции \(HTTP API, IIDK, ActiveX, HTTP Сервер, Axxon Next\)](#).

12.1.5.8.2 Получение архивного видео

Для получения архивного видео с Камеры 1 следует отправить на порт 900 следующие сообщения:

CAM|1|ARCH_FRAME_TIME|time<dd-mm-yy HH:MM:SS.FFF> – для установки времени, начиная с которого требуется просматривать архив.

CAM|1|PLAY|compress<> – для получения архивного видео. Работа с архивным видео осуществляется таким же образом, как с живым.

12.1.5.8.3 Получение списка временных интервалов

Для того, чтобы получить список временных интервалов, содержащих видеозаписи за определенную дату, необходимо послать на порт 900 следующее сообщение:

CAM|id|ARCH_GET_INTERVALSREC|date<>,time_with_milliseconds<1>,with_filenames<1>

Параметр date<> может принимать значение date<dd-mm-yy> или быть оставлен пустым. В первом случае будут запрошены интервалы за указанную дату, во втором – даты, за которые присутствует архив.

Если задан параметр time_with_milliseconds<1>, то интервалы придут с миллисекундами (например, 14:08:55.677 14:09:55.641). Если задан параметр with_filenames<1>, то сообщение будет содержать названия файлов (например, 'C:\VIDEO\26-04-21 14\0_01'). Эти параметры не являются обязательными.

В результате будет получено сообщение вида

Event: CAM|id|SET_INTERVALSREC|intervals<>,date<>,timezone<>

Значение параметра intervals<> имеет следующий вид: intervals<begin1 end1\nbegin2 end2...\nbeginN endN|date1\ndate2...\ndateN\n>

Время начала и время конца разделяется одним пробелом (код 0x20), интервалы отделяются друг от друга символом переноса строки '\n' (код 0x0A).

- begin1, begin2, ... beginN – времена начал интервалов в формате HH:MM:SS (возвращается, если запрошена точная дата).

- end1, end2, ... endN – времена концов интервалов в формате HH:MM:SS(возвращается, если запрошена точная дата).
- date1, date2, ... dateN – даты для которых присутствуют записи в архиве (возвращается, если поле date в запросе пусто или отсутствует).

Параметр date<dd-mm-yy> – это дата, за которую запрашивались интервалы, или пустое значение (date<>), если запрашивались даты за весь период.

Параметр timezone<> определяет смещение времени клиента (IIDK) относительно времени Сервера в минутах.

Примеры:

- Если Сервер во временной зоне -1 UTC, то в ответ на команду CAM|1|ARCH_GET_INTERVALSREC| будет получено событие с параметром timezone<60>
- Если Сервер во временной зоне +3 UTC, то в ответ на команду CAM|1|ARCH_GET_INTERVALSREC| будет получено событие с параметром timezone<-180>

Пример запроса:

```
CAM|1|ARCH_GET_INTERVALSREC|date<26-04-21>,time_with_milliseconds<1>,with_filenames<1>
```

Пример ответа:

```
CAM|1|SET_INTERVALSREC|time_with_milliseconds<1>,intervals<14:07:55.659 14:08:55.637 - - file:'
C:\VIDEO\26-04-21 14\0._01'
14:08:55.677 14:09:55.641 - - file:'C:\VIDEO\26-04-21 14\1._01'
14:09:55.681 14:10:35.667 - - file:'C:\VIDEO\26-04-21 14\2._01'
14:17:12.444 14:18:09.553 - - file:'C:\VIDEO\26-04-21 14\3._01'
14:29:41.132 14:29:41.292 - - file:'C:\VIDEO\26-04-21 14\4._01'
14:29:41.432 14:29:51.363 - - file:'C:\VIDEO\26-04-21 14\5._01'
14:34:38.788 14:35:03.117 - - file:'C:\VIDEO\26-04-21 14\6._01'
14:35:03.267 14:36:19.151 - - file:'C:\VIDEO\26-04-21 14\7._01'
>,timezone<-180>,module<iidk_client_test_x64.exe>,_TRANSPORT_ID<>,with_filenames<1>,date<26-04-2
1>,slave_id<VDESKTOP.2A180FE5-BA8E-420C-B80E-90DC20516A26>,durationMS<0>
```

12.1.5.9 Управление телеметрией

Управление телеметрией через IIDK осуществляется при помощи обычных реакций, описанных в [Руководстве по программированию](#) в разделе [TELEMETRY Поворотное устройство](#), например:

CORE||DO_REACT|

source_type<TELEMETRY>,source_id<1.1>,action<LEFT>,params<1>,param0_name<tel_prior>,param0_val<3> – сообщение на порт 1030 для поворота объектива камеры влево с высоким приоритетом.

TELEMETRY|1.1|LEFT|speed<2>,tel_prior<3> – реакция на порт 1030 для поворота объектива камеры влево с высоким приоритетом и средней скоростью.

12.1.5.10 Операции со слоем карты

Команда задания размера и положения значка объекта **Камера 1** на слое 1 выполняется одним из следующих способов:

1. Посылкой сообщения на порт 1030 **CORE||DO_REACT|**
source_type<MAPLAYER>,source_id<1>,action<CUSTOMIZE_OBJECT>,params<7>,param0_name<x>,param0_val<200>,param1_name<y>,param1_val<200>,param2_name<objtype>,param2_val<CAM>,param3_name<objid>,param3_val<1>,param4_name<a>,param4_val<90>,param5_name<w>,param5_val<70>,param6_name<h>,param6_val<80>

Здесь x, y, w, h – координаты и размер значка объекта на карте.
a – угол наклона значка.

2. Посылкой реакции на порт 1030

MAPLAYER|1|CUSTOMIZE_OBJECT|x<200>,y<200>,objtype<CAM>,objid<1>,a<90>,w<70>,h<80>

Вывод слоя 1 в окне интерактивной карты осуществляется одним из следующих способов:

1. Посылкой сообщения на порт 1030: **CORE||DO_REACT|source_type<MAPLAYER>,source_id<1>,action<ACTIVATE>**
2. Посылкой реакции на порт 1030: **MAPLAYER|1|ACTIVATE-**

12.1.5.11 Получение информации об очередях ядра командой GET_QUEUE_INFO

Для получения информации об очередях ядра используется команда GET_QUEUE_INFO:

CORE|GET_QUEUE_INFO

Примечание.

Если в системе несколько объектов **Интерфейс IIDK**, также может быть указан параметр receiver_id – см. [Особенности работы с системой в многопользовательском режиме](#).

В ответ будет получена строка вида:

ACTIVEV|11|QUEUE_INFO|thread2<0>,thread1<0>,thread0<0>,posted_events<0>,_TRANSPORT_ID<>,server_reacts<0>,posted_reacts<0>,events_inwork<0>,coremanager_events<0>,thread3<0>

Параметры ответа соответствуют информации, отображаемой в окне **Статистика очередей** (открывается по нажатию Alt+F2). Описание параметров:

threadN<> – количество элементов в очереди потока N.

posted_events<> – количество входящих событий.

posted_reacts<> – количество реакций в работе.

coremanager_events<> – количество событий на отправку.

server_reacts<> – количество реакций на отправку.

events_inwork<> – количество событий в работе.

12.1.5.12 Проигрывание аудиоархива за период. Команда START_PLAY_TIME

Для проигрывания аудиоархива заданного микрофона за определенный период с помощью конкретного объекта **Динамик** следует выполнить команду вида

SPEAKER|{id}|START_PLAY_TIME|speaker_id<>,mic_id<>,time_start<>,time_end<>,cam_id<>

Примеры:

SPEAKER|1|START_PLAY_TIME|speaker_id<1>,mic_id<2>,time_start<28-02-20 14:23:24.092>,time_end<28-02-20 14:23:27.092>

SPEAKER|1|START_PLAY_TIME|speaker_id<1>,mic_id<2>,time_start<02-03-20 12:01:24.092>,time_end<02-03-20 12:04:27.092>,cam_id<1>

Если указан идентификатор камеры **cam_id<>**, то будет проигрываться аудиоархив, записанный синхронно с видеоархивом (т.е. из папки VIDEO). Если параметр cam_id<> не указан, то проигрывается архив из папки AUDIO.

speaker_id<> – идентификатор объекта Динамик

mic_id<> – идентификатор объекта Микрофон

time_start<> – время начала отрезка аудиоархива

time_end<> – время окончания отрезка аудиоархива

12.1.5.13 Воспроизведение тонового набора. Команды START_TONE и STOP_TONE

Для воспроизведения цифр, букв и символов в тоновом наборе с помощью конкретного объекта **Динамик** следует выполнить команду START_TONE:

```
SPEAKER|{id}|START_TONE|speaker_id<>,symbols<>,duration<>
```

Параметры команды:

- **speaker_id<>** – идентификатор объекта **Динамик**, на котором будет воспроизведение;
- **symbols<>** – символы, которые надо озвучить. Символами могут быть только 1234567890*#ABCD, остальные будут игнорироваться. Регистр для букв не имеет значения;
- **duration<>** – продолжительность озвучивания каждого символа в миллисекундах (мс).

Для досрочного прекращения воспроизведения используется команда STOP_TONE:

```
SPEAKER|{id}|STOP_TONE|speaker_id<>
```

В ней **speaker_id<>** – идентификатор объекта **Динамик**, на котором нужно остановить воспроизведение.

Примеры команд:

```
SPEAKER|1|START_TONE|speaker_id<1>,symbols<12>,duration<200>
SPEAKER|1|STOP_TONE|speaker_id<1>
```

12.1.5.14 Получение статистики по видеопотоку. STATISTIC

Для вывода статистики по видеопотоку используются следующие команды:

- **STATISTIC||GET** – получить статистику по видео;
- **STATISTIC||START|interval<>** – начать отправку статистики с заданным в секундах интервалом (например, interval<10>);
- **STATISTIC||STOP** – остановить отправку статистики.

Команды отправляются через 900 порт. Примеры полученных сообщений со статистикой:

```
STATISTIC|[MONITOR][2][CAM][9][IN]|SET|count<20>,bps<1.42042e+007>,fps<19.728>,_TRANSPORT_ID<>
STATISTIC|[MONITOR][2][CAM][9][OUT]|SET|count<20>,bps<888264>,fps<19.7392>,_TRANSPORT_ID<>
```

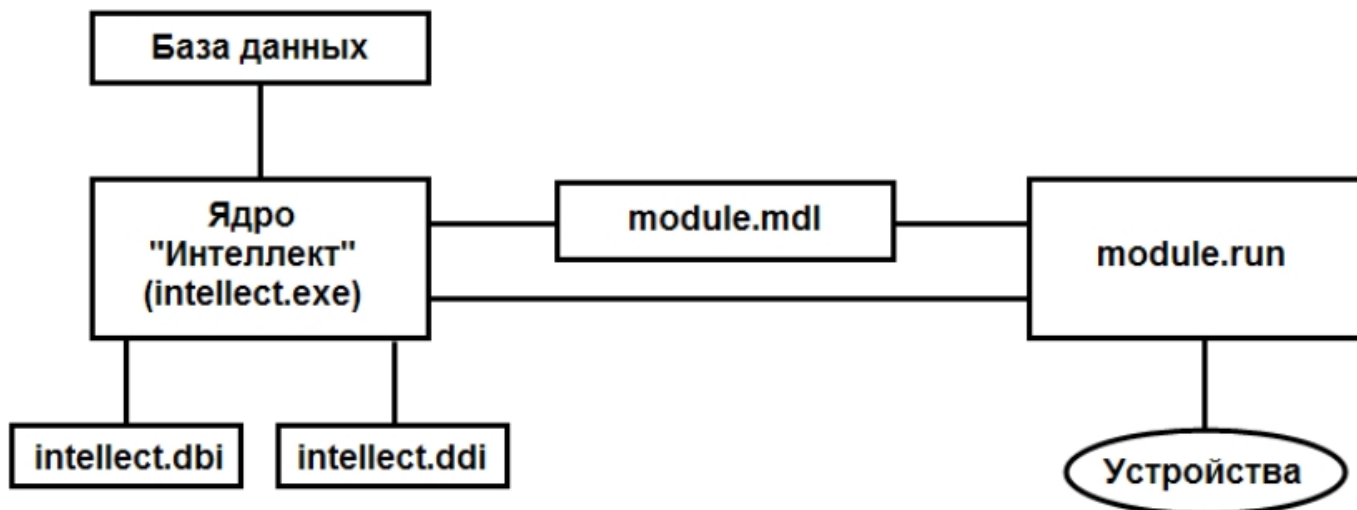
12.2 Интеграция аппаратно-программных модулей с ПК Интеллект

12.2.1 Общие сведения об интеграции аппаратно-программных модулей

Процесс интеграции аппаратно-программных (функциональных) модулей с ПК *Интеллект* состоит из следующих этапов:

1. Редактирование DBI-файла.
2. Редактирование DDI-файла.
3. Подготовка файла module.mdl, где module – имя интегрируемого модуля (данный файл является преобразованным DLL-файлом).
4. Подготовка исполнительного файла module.run, где module – имя интегрируемого модуля (этот файл является преобразованным exe-файлом).
5. Размещение module.mdl и module.run в каталоге *Интеллект\Modules*.

Схема взаимодействия функционального модуля с ядром системы представлена на рисунке.



DBI- и DDI-файлы содержат необходимую для функционирования ядра системы информацию об интегрированных функциональных модулях (объектах). DBI-файл содержит описание структуры конфигурационной базы данных ПК *Интеллект*. В DDI-файле хранится описание объектов и их параметров. При интеграции объекта в данные файлы заносят наименование, параметры интегрируемого объекта, связанные с ним системные события и реакции.

MDL-файлы обеспечивают работу с объектами одного типа: создание, изменение, удаление, изменение при настройке или в процессе работы параметров объекта и сохранение их в базе данных, выполнение некоторых специализированных операций с объектом. Также MDL-файл обеспечивает пересылку параметров созданных или измененных объектов исполняемому модулю (RUN-файлу) и хранит конфигурации настроечных панелей объектов.

Исполняемый RUN-файл осуществляет взаимодействие с устройствами, транслирует в ядро информацию о событиях, обеспечивает выполнение управления устройствами.

Далее описываются этапы интеграции модулей на примере демонстрационного модуля [DEMO](#), эмулирующего работу с виртуальным оборудованием. Данный модуль включает в себя устройства с уникальными адресами для обращения к этим устройствам и их опроса. В системе возможно выполнение определенного набора действий с устройствами и передача всех происходящих в них событий ядру системы.

12.2.2 Редактирование DBI-файла

Файл `intellect.dbi` содержит основной перечень таблиц и полей базы данных. Рекомендуется создавать собственный шаблон базы данных в отдельном файле – `intellect.xxx.dbi`, где `xxx` – уникальная часть имени файла. Использование отдельного файла позволяет избежать повторного включения таблиц и полей в случае обновления ПК *Интеллект*. При запуске программного комплекса DBI-файлы объединяются.

12.2.2.1 Добавление объектов в `intellect.dbi`

Добавление объектов в `intellect.dbi` выполняется следующим образом:

1. Открыть в текстовом редакторе файл `intellect.dbi`, расположенный в корневом каталоге ПК *Интеллект*.
2. Добавить в `intellect.dbi` объекты. Для этого необходимо в квадратных скобках указать имя, используемое для идентификации объекта, и далее объявить его поля. Синтаксис объявления полей имеет вид:

<Имя поля>, <Тип> [, <Размер>]

Примечание.

Размер задается только полям с типом **CHAR**.

В таблице приведены обязательные поля для всех объектов ПК *Интеллект*.

Поле	Описание
id	Уникальный идентификатор объекта
name	Имя объекта
parent_id	Идентификатор родительского объекта
flags	Параметр для внутренних нужд системы

**Внимание!**

Поле **flags** не может использоваться внешними приложениями.

Допустимые типы данных описаны в таблице.

Тип данных	Описание
BIT	Используется для создания поля-флажка, принимающего логические значения «Да» или «Нет»
CHAR	Используется для полей, заполняемых небольшим количеством символов
DATETIME	Используется для полей, в которые вводятся дата и время. Маска для даты – гггг-мм-дд, для времени – чч:мм:сс.xxx
DOUBLE	Используется для полей, содержащих числа с плавающей запятой
INTEGER	Используется для полей, содержащих целые числа
TEXT	Используется для полей, содержащих текстовые строки

Для объектов демонстрационного модуля *DEMO*, кроме обязательных полей, добавлены поля:

- a. **port** – адрес COM-порта;
- b. **address** – адрес устройства.

Результат добавления объектов и объявления полей в intellect.dbi представлен на рисунке.

```

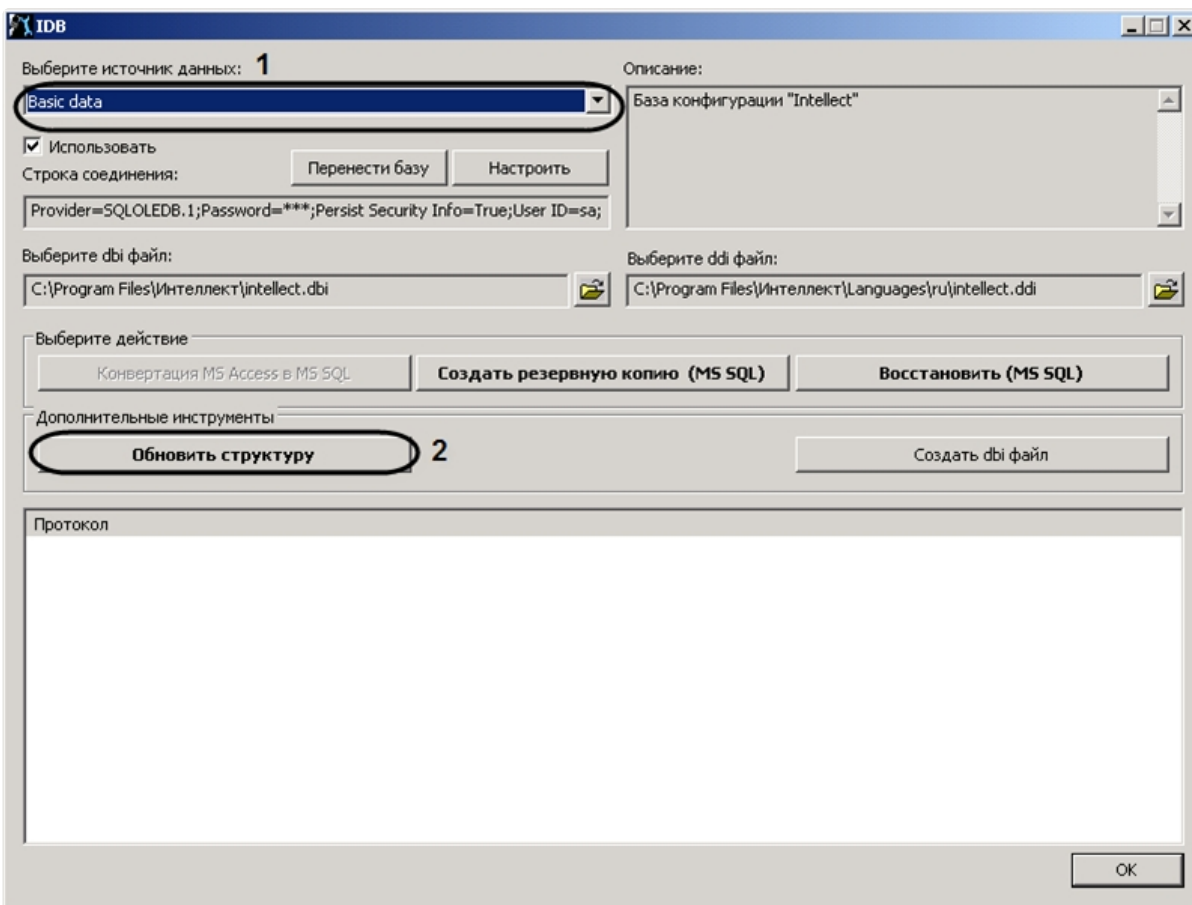
x, INTEGER
y, INTEGER
w, INTEGER
h, INTEGER
exe, CHAR, 255
guid, UNIQUEIDENTIFIER

[OBJ_ZONE]
id, CHAR, 40
name, CHAR, 60
parent_id, CHAR, 40
Flags, INTEGER
guid, UNIQUEIDENTIFIER

[OBJ_DEMO]
id, CHAR, 16
name, CHAR, 60
parent_id, CHAR, 16
port, CHAR, 5
Flags, INTEGER

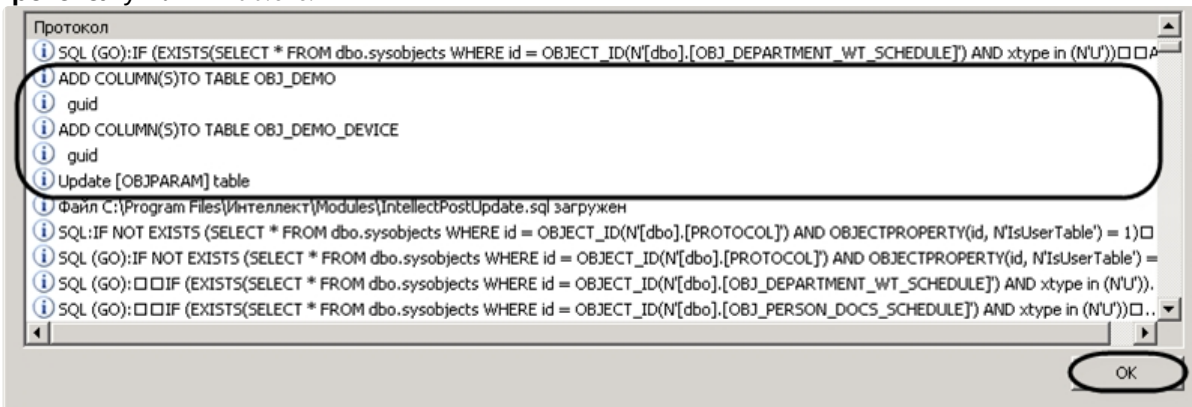
[OBJ_DEMO_DEVICE]
id, CHAR, 16
name, CHAR, 60
parent_id, CHAR, 16
address, INTEGER
Flags, INTEGER
  
```

3. Сохранить изменения в файле intellect.dbi.
4. Запустить утилиту *idb.exe*, расположенную в корневом каталоге ПК *Интеллект*.



5. Из списка **Выберите источник данных:** выбрать **Basic data** (1).
6. Нажать кнопку **Обновить структуру** (2).
Будет запущен процесс обновления структуры базы данных. Выполнение процесса отображается в окне

Протокол утилиты *idb.exe*.



7. Нажать кнопку **OK** для завершения работы с утилитой *idb.exe*.

В результате обновления структуры будут созданы таблицы в базе конфигурации *Intellect*.

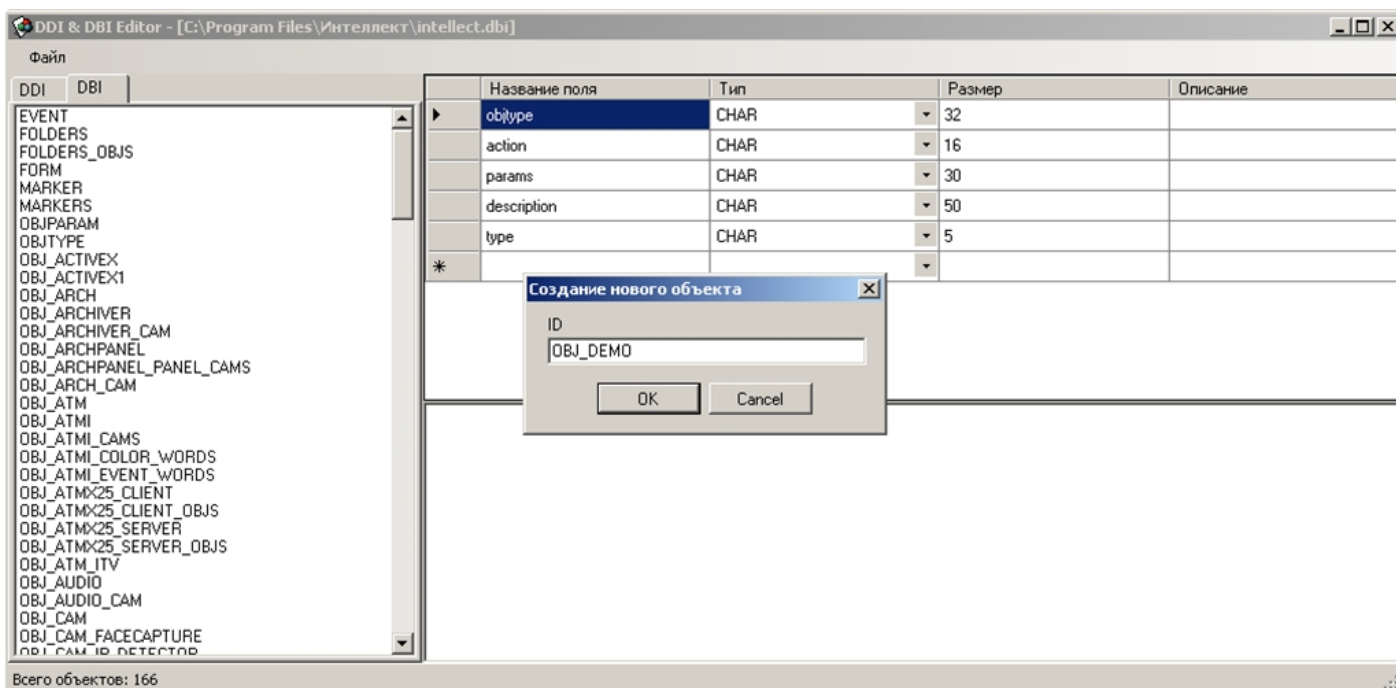
12.2.2.2 Использование утилиты *ddi.exe* для работы с DBI-файлами

Для добавления объекта в DBI-файл с помощью утилиты *ddi.exe* необходимо выполнить следующие действия:

1. Запустить утилиту *ddi.exe*, расположенную в каталоге *Интеллект\Tools*.
2. В окне утилиты перейти на вкладку **DBI**.
3. В меню **Файл** выбрать пункт **Открыть**. В результате выполнения операции появится диалоговое окно **Открыть**.
4. Выбрать файл *intellect.dbi*, расположенный в корне директории установки ПК *Интеллект*. В утилите *ddi.exe* отобразится список объектов.
5. В контекстном меню списка объектов выбрать пункт **Добавить** для добавления нового объекта.

Примечание.
Также возможно добавить объект с помощью клавиши **Insert**.

6. В открывшемся диалоговом окне ввести имя, используемое для идентификации объекта, в поле **ID** и нажать **OK**.



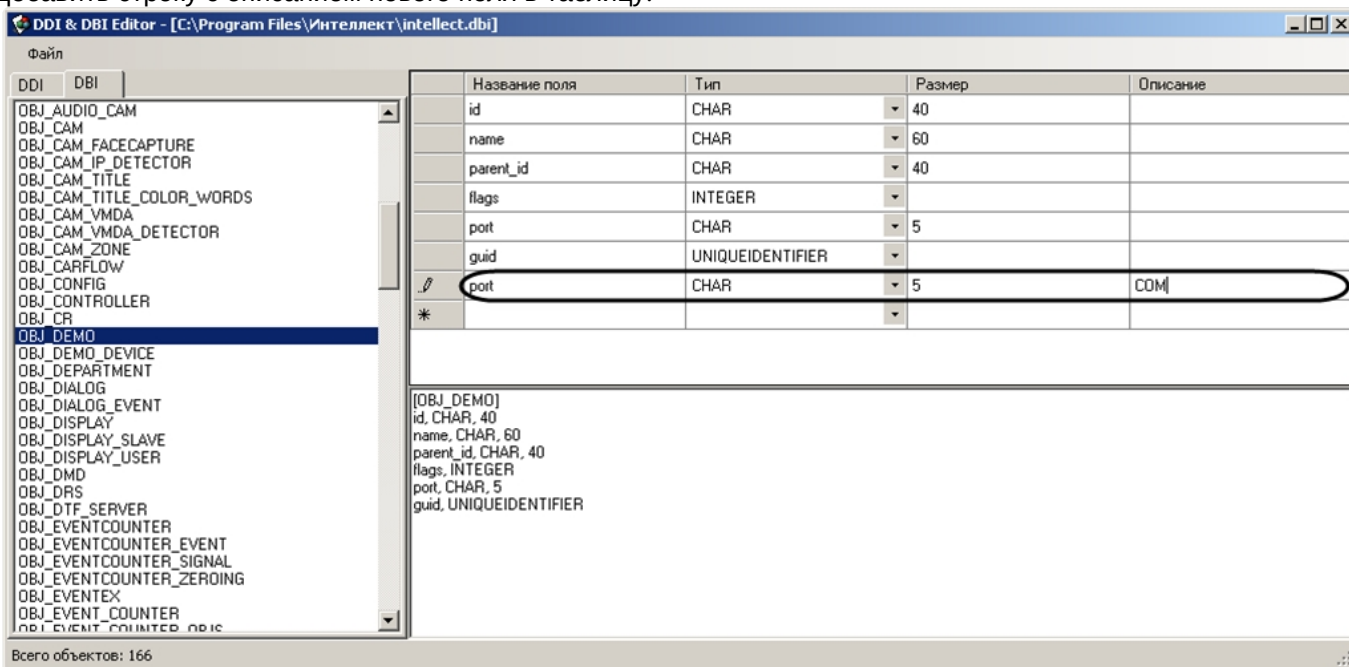
Примечание.

Созданному объекту будут автоматически добавлены обязательные поля (см. раздел [Добавление объектов в intellect.dbi](#)).

Добавление объекта в DBI-файл завершено.

Добавление поля выполняется следующим образом:

1. Выбрать созданный объект в левой части окна утилиты ddi.exe
2. Добавить строку с описанием нового поля в таблицу.



3. Сохранить внесенные изменения, выбрав в меню **Файл** пункт **Сохранить**.

Добавление поля выполнено.

Внимание!

После изменения DBI-файлов требуется обновить структуру базы данных с помощью утилиты idb.exe (см. раздел [Добавление объектов в intellect.dbi](#)).

12.2.3 Редактирование DDI-файла

DDI-файл представляет собой файл в формате xml, который содержит следующую информацию об объектах:

1. Реакции – действия, которые могут выполнять объекты.
2. События, которые могут генерировать объекты.
3. Состояния, в которых могут находиться объекты.
4. Правила перехода объектов из одного состояния в другое по определенным событиям.
5. Имена bmp-файлов, используемых для отображения объектов на *Карте*.

Файл intellect.ddi содержит свойства основных объектов ПК *Интеллект*. Для собственных объектов рекомендуется создавать отдельный файл – intellect.xxx.ddi, где xxx – уникальная часть имени файла. Использование отдельного файла позволяет избежать повторного включения свойств объектов в случае обновления ПК *Интеллект*. При запуске программного комплекса DDI-файлы объединяются.

В случае, если объект дублируется в нескольких ddi-файлах, при запуске ПК *Интеллект* применяются свойства объекта из последнего файла в соответствии с сортировкой файлов по имени. Например, если какой-либо объект

описан в файлах `intellect.xxx.ddi`, `intellect.xxx1.ddi` и `intellect.xxx2.ddi`, будут применены свойства из файла `intellect.xxx2.ddi`.

12.2.3.1 Добавление в `intellect.ddi` информации об объекте

В данном разделе приведен пример добавления в `intellect.ddi` информации об объекте **DEMO** с использованием текстового редактора.

Для добавления информации об объекте **DEMO** в `intellect.ddi` необходимо выполнить следующие действия:

1. Открыть в текстовом редакторе файл `intellect.ddi`, расположенный в каталоге `Интеллект\Languages\ru`.
2. В раздел `<DataSetDDI>` добавить дочерний элемент `<Objects>`, содержащий описание объекта.

```
<Objects>
  <ObjectName>DEMO</ObjectName>
  <VisibleName>Демо</VisibleName>
  <Events>
    <EventName>LOST</EventName>
    <EventDescription>Потеря связи</EventDescription>
    <IsSoundEnabled>>false</IsSoundEnabled>
    <IsNetworkDisabled>>false</IsNetworkDisabled>
    <IsProtocolDisabled>>false</IsProtocolDisabled>
    <IsWindowsLogEnabled>>false</IsWindowsLogEnabled>
  </Events>
  <Events>
    <EventName>RESTORE</EventName>
    <EventDescription>Восстановление связи</EventDescription>
    <IsSoundEnabled>>false</IsSoundEnabled>
    <IsNetworkDisabled>>false</IsNetworkDisabled>
    <IsProtocolDisabled>>false</IsProtocolDisabled>
    <IsWindowsLogEnabled>>false</IsWindowsLogEnabled>
  </Events>
  <Icons>
    <FileName>demo</FileName>
    <IconName>demo</IconName>
  </Icons>
  <States>
    <StateName>DETACHED</StateName>
    <ImgName>detached</ImgName>
    <IsStateFlashing>>false</IsStateFlashing>
  </States>
  <States>
    <StateName>NORMAL</StateName>
    <ImgName>normal</ImgName>
    <IsStateFlashing>>false</IsStateFlashing>
  </States>
  <Rules>
    <EventName>RESTORE</EventName>
    <FromStateName>DETACHED</FromStateName>
    <ToStateName>NORMAL</ToStateName>
  </Rules>
  <Rules>
    <EventName>LOST</EventName>
    <FromStateName>NORMAL</FromStateName>
    <ToStateName>DETACHED</ToStateName>
  </Rules>
</Objects>
```

Примечание

Для объекта **DEMO** отсутствует раздел `<Reacts>`, так как данный объект не выполняет никаких действий.

Примечание

Элементы DDI-файла подробно описаны в разделе [ПРИЛОЖЕНИЕ 1. Описание структуры ddi-файла](#)

3. Сохранить изменения в файле intellect.ddi.

Внесение в intellect.ddi информации об объекте **DEMO** завершено.

Внимание!

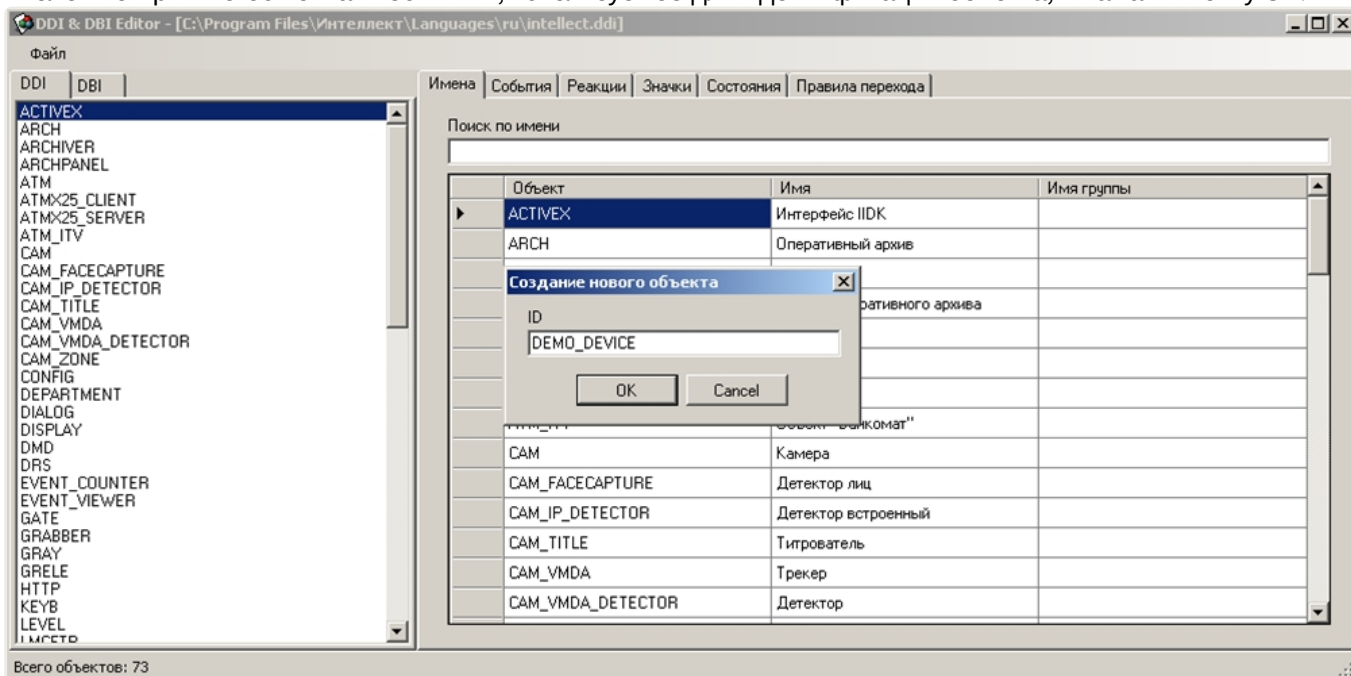
После изменения DDI-файлов требуется обновить структуру базы данных с помощью утилиты idb.exe (см. шаги 4-7 раздела [Добавление объектов в intellect.dbi](#)).

12.2.3.2 Использование утилиты ddi.exe для работы с DDI-файлами

В данном разделе приведен пример добавления в intellect.ddi информации об объекте **DEMO_DEVICE** с использованием утилиты *ddi.exe*.

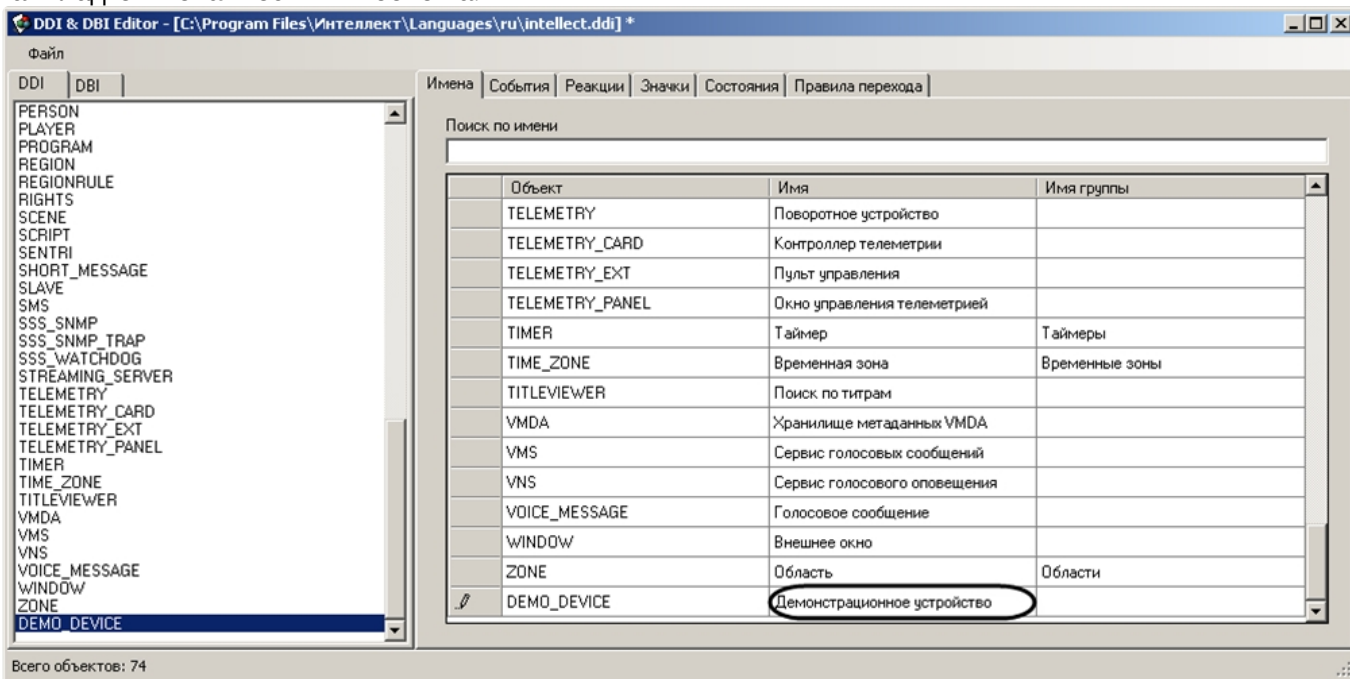
Для добавления в intellect.ddi информации об объекте **DEMO_DEVICE** необходимо выполнить следующие действия:

1. Открыть файл intellect.ddi в утилите одним из следующих способов:
 - a. Из утилиты ddi.exe:
 - i. Запустить утилиту *ddi.exe*, расположенную в каталоге *Интеллект\Tools*.
 - ii. В окне утилиты перейти на вкладку **DDI**.
 - iii. В меню **Файл** выбрать пункт **Открыть**. В результате выполнения операции появится диалоговое окно **Открыть**.
 - iv. Выбрать файл intellect.ddi, расположенный в каталоге *Интеллект\Languages\ru*. В утилите *ddi.exe* отобразится список объектов.
 - b. Открыть каталог *Интеллект\Languages\ru* в проводнике Windows или любой другой программе для управления файлами и дважды щелкнуть по файлу intellect.ddi.
2. Добавить объект, выбрав в контекстном меню списка объектов пункт **Добавить** или нажав кнопку **Insert**.
3. В поле **ID** открывшегося окна ввести имя, используемое для идентификации объекта, и нажать кнопку **OK**.



В результате выполнения операции объект **DEMO_DEVICE** отобразится в списке объектов.

4. На вкладке **Имена** ввести имя объекта.



5. Добавить информацию об объекте **DEMO_DEVICE** на соответствующих вкладках.

a. Добавить события **ON** и **OFF** на вкладке **События**.

Имена	События	Реакции	Значки	Состояния	Правила перехода		
	Название	Описание	Обработка сообщений	Звуковая поддержка	Не слать в сеть	Не протоколировать	Журнал Windows
	ON	Включено	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	OFF	Выключено	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
*			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

b. Добавить действия **ON** и **OFF** на вкладке **Реакции**.

Имена	События	Реакции	Значки	Состояния	Правила перехода
	Реакция	Описание	Постановка раздела на охрану		
	ON	Включить	<input type="checkbox"/>		
	OFF	Выключить	<input type="checkbox"/>		
*			<input type="checkbox"/>		

c. На вкладке **Значки** указать часть имени bmp-файла, которая является идентификатором изображения. Идентификатор изображения позволяет использовать несколько bmp-файлов для представления на *Карте* объектов одного типа.

Имена	События	Реакции	Значки	Состояния	Правила перехода
	Имя файла	Название			
	demo_device	Модуль DEMO			
*					

d. На вкладке **Состояния** добавить состояния **ON** и **OFF**. Для отображения состояния объекта на *Карте* необходимо указать часть имени, которая является идентификатором состояния, соответствующего bmp-файла.

Имена	События	Реакции	Значки	Состояния	Правила перехода
	Название	Изображение	Описание	Мерцание при тревоге	
	ON	on	Включено	<input type="checkbox"/>	
	OFF	off	Выключено	<input type="checkbox"/>	
*				<input type="checkbox"/>	

Примечание.

Каталог Интеллект\Bmp должен содержать bmp-файлы, имена которых составлены следующим образом:

<Идентификатор изображения>_<Идентификатор состояния>

Если идентификатор изображения не задан, то имя bmp-файла должно иметь вид:

<Идентификатор объекта>_<Идентификатор состояния>

Примечание

Объекты на Карте могут быть отображены с помощью линий, т.е. без использования bmp-файлов. В этом случае, если изменяется состояние объекта, меняется цвет линии. Цвет (RGB) состоянию задается следующим образом:

<Состояние>\$R:G:B

- е. На вкладке **Правила перехода** задать правило перехода из одного состояния в другое по определенному событию.

Имена	События	Реакции	Значки	Состояния	Правила перехода	
	Событие				Переход из состояния	Переход в состояние
	OFF			ON	OFF	
	ON			OFF	ON	
*						

Примечание.

Если поле **Переход из состояния** оставить пустым, то переход будет осуществляться из любого состояния.

6. Сохранить изменения, выбрав в меню **Файл** пункт **Сохранить**.

Информация об объекте **DEMO_DEVICE** внесена.

Примечание.

Поля таблиц утилиты ddi.exe подробно описаны в [ПРИЛОЖЕНИЕ 1. Описание структуры ddi-файла](#)

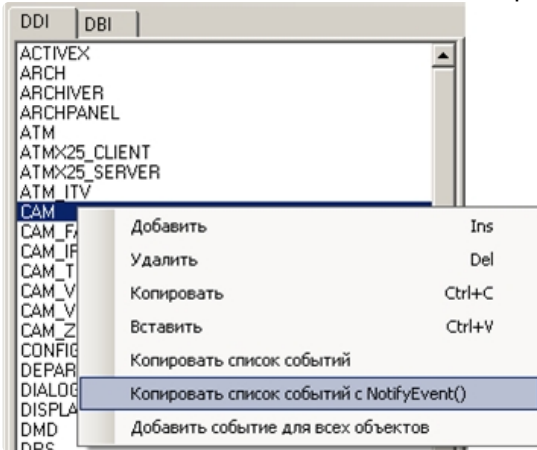
Внимание!

После изменения DDI-файлов требуется обновить структуру базы данных с помощью утилиты idb.exe (см. шаги 4-7 раздела [Добавление объектов в intellect.dbi](#)).

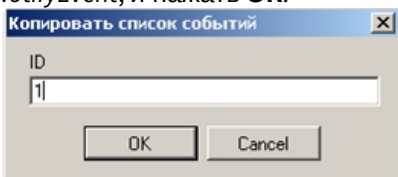
12.2.4 Дополнительные возможности утилиты ddi.exe

Утилита *ddi.exe* представляет собой удобный инструмент для удаления, добавления, редактирования и копирования в буфер обмена свойств объекта (событий, реакций и т.д.). Дополнительно утилита позволяет копировать в буфер обмена события объекта в виде параметра функции *NotifyEvent*. Для этого необходимо выполнить следующие действия:

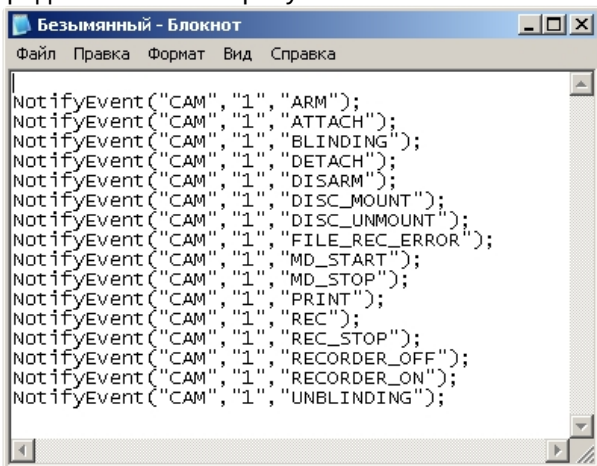
1. В контекстном меню списка объектов выбрать пункт **Копировать список событий с NotifyEvent()**.



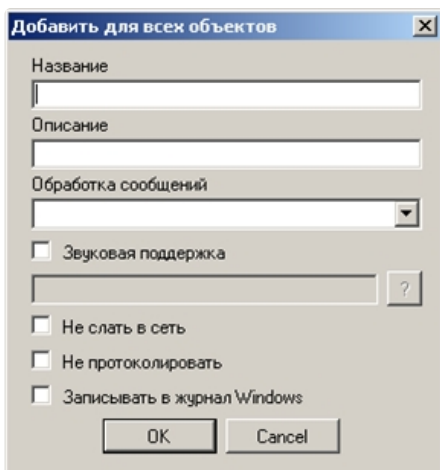
2. В открывшемся окне ввести идентификационный номер объекта, который следует использовать в функции *NotifyEvent*, и нажать **ОК**.



Копирование списка событий завершено. Буфер обмена будет содержать события объекта в виде, представленном на рисунке.



В случае, если требуется добавить событие всем объектам, следует в контекстном меню выбрать пункт **Добавить событие для всех объектов**. В результате выполнения операции будет открыто диалоговое окно **Добавить для всех объектов**, в котором параметрам создаваемого события задаются значения.



Для добавления объектов из других DBI- и DDI-файлов следует в меню **Файл** выбрать пункт **Вставить из файла**.

12.2.5 Разработка MDL-файла

Для создания mdl-файла необходимо использовать два класса:

1. *NissObjectDLLExt*. Все объекты наследуются от этого класса с переопределением его виртуальных методов.
2. *CoreInterface*. Методы класса используются для получения параметров объектов системы.

Объявленные классы и методы содержатся в заголовочном файле *nissdll.h*. Код, содержащийся в файле *nissdll.h*, представлен в разделе [ПРИЛОЖЕНИЕ 2. Объявление классов *NissObjectDLLExt* и *CoreInterface*](#).

Примечание.

Под методами класса подразумеваются процедуры и функции, объявленные в теле класса.

Описание методов класса *NissObjectDLLExt* приведено в таблице.

Метод	Описание	Пример
CoreInterface* m_pCore	Указатель на интерфейс ядра	
virtual BOOL IsWantAllEvents()	Возвращает TRUE, если необходимо в функции OnEvent получать события от всех объектов, FALSE - если только от своего объекта.	если указать "CAM,GRABBER", то при изменении настроек этих объектов для объекта DEMO придут сообщения: DEMO 1 UPDATE_CAM параметры камеры
virtual CString DescribeSubscribeObjectsList()	Через запятую указываются типы объектов, при изменении которых происходит уведомление текущего объекта	DEMO 1 UPDATE_GRABBER параметры устройства видеоввода
virtual CString GetObjectType()	Возвращает тип объекта	<pre>virtual CString GetObjectType() { return "DEMO"; }</pre>
virtual CString GetParentType()	Возвращает тип родительского объекта	<pre>virtual CString GetParentType() { return "SLAVE"; }</pre>

Метод	Описание	Пример
virtual int GetPos()	<p>Возвращает позицию объекта в ключевом файле "intellect.sec".</p> <p>Внимание! Этот параметр должен быть согласован с компанией «Ай Ти Ви групп»</p>	<pre>virtual int GetPos() { return -1; }</pre> <p><i>Примечание. При запуске ПК Интеллект в демо-режиме функция возвращает -1</i></p>
virtual CString GetPort()	<p>Возвращает номер порта, через который будет происходить соединение и обмен сообщениями между объектом и ядром</p> <p>Внимание! Этот параметр должен быть согласован с компанией «Ай Ти Ви групп»</p>	<pre>virtual CString GetPort() { return "1100"; }</pre>
virtual CString GetProcessName()	<p>Возвращает имя процесса. Используется ядром для поиска и автоматического запуска исполнительного модуля при старте системы и инициализации модуля</p>	<pre>virtual CString GetProcessName() { return "demo"; }</pre>
virtual CString GetDeviceType()	<p>Определяет тип объекта и характер его поведения.</p> <p>ACD – объект этого типа получает все события, связанные с созданием, изменением и удалением следующих объектов: Пользователи, Временная зона и Уровни доступа</p> <p>ACD2– тип аналогичный ACD с дополнительной (обеспеченной ядром) возможностью автоматически удалять временные (с ограниченным сроком действия) карточки</p> <p>Тип ACR указывает на то, что объект является считывателем</p>	<p>Все объекты типа ACR доступны в раскрывающемся списке Точка прохода</p>
virtual BOOL HasChild()	<p>Возвращает TRUE, если у объекта есть дочерние объекты, иначе – FALSE</p>	<pre>virtual BOOL HasChild() { return TRUE; }</pre>
virtual UINT HasSetupPanel()	<p>Если у объекта имеется панель настроек, метод возвращает TRUE, иначе – FALSE</p>	<pre>virtual UINT HasSetupPanel() { return TRUE; }</pre>
virtual void OnPanellnit(CWnd*)	<p>Используется при инициализации панели настроек объекта. В качестве параметра передается указатель на окно панели настроек</p>	

Метод	Описание	Пример
virtual void OnPanelLoad(CWnd*,Msg&)	Используется при загрузке панели настроек для получения параметров объекта. В качестве параметра метода передается указатель на окно панели настроек и ссылка на сообщение, через которое осуществляется передача параметров объекта, и заполнение ими необходимых полей в панели настроек	<pre> virtual void OnPanelLoad(CWnd* pwnd,Msg& params) { CString s; s = arams.GetParam("port"); pwnd->GetDlgItem(IDC_PORT)-> SetWindowText(s); } </pre>
virtual void OnPanelSave(CWnd*,Msg&)	Используется при выгрузке панели настроек для сохранения параметров объекта. В качестве параметра метода передается указатель на окно панели настроек и ссылка на сообщение, через которое осуществляется передача параметров объекта и сохранение их в БД	<pre> virtual void OnPanelSave(CWnd* pwnd,Msg& params) { CString s; pwnd-> GetDlgItem(IDC_PORT)-> GetWindowText(s); params.SetParam("port",s); } </pre>
virtual void OnPanelExit(CWnd*)	Используется при закрытии панели настроек объекта. В качестве параметра передается указатель на окно панели настроек	
virtual void OnPanelButtonPressed(CWnd*,UI NT)	<p>Предназначен для обработки нажатий кнопок на панели настроек объекта. В качестве параметра передается указатель на окно панели настроек и идентификатор кнопки.</p> <p><i>Примечание. Числовые значения идентификаторов кнопок должны начинаться с номера 1151. Например, для кнопки Test идентификатор в файле <i>Resource.h</i> определяется как:</i></p> <p>#define IDC_TEST 1151</p>	<pre> Virtual void OnPanelButtonPressed (CWnd* pwnd,UINT id) { if(id==IDC_TEST) { React react("DEMO",Id,"TEST"); m_pCore->DoReact(react); } } </pre>

Метод	Описание	Пример
	<p>Если необходимо по нажатию кнопки вывести собственное диалоговое окно, созданное в этом же MDL-файле, то необходимо предварительно использовать код, указанный в примере</p>	<pre>HINSTANCE prev_hinst = AfxGetResourceHandle(); HMODULE hRes = GetModuleHandle("demo.mdl"); If (hRes) AfxSetResourceHandle (hRes); //Код вывода диалогового окна: CXXXDialog dlg; dlg.DoModal(); AfxSetResourceHandle(prev_hinst);</pre>
virtual BOOL IsRegionObject()	<p>Указывает на то, что объект будет поддерживать разделы ПК <i>Интеллект</i>. Разделы применяются для группировки объектов. Также они могут использоваться в системе отчетов</p>	
virtual BOOL IsProcessObject()	<p>Указывает на то, что объект будет поддерживать возможность одновременного запуска и параллельной работы нескольких исполнительных модулей. Например, это может использоваться для запуска отдельного модуля непосредственно для каждого COM-порта.</p> <p><i>Примечание. Рекомендуется использовать один рабочий модуль. Это упростит отладку и модификацию модуля</i></p>	
virtual void OnCreate(Msg&)	<p>Используется при создании объекта. В качестве параметра передается ссылка на сообщение, содержащее информацию по объекту. Здесь же указываются параметры по умолчанию</p>	<pre>virtual void OnCreate (Msg& msg) { msg.SetParam ("port","COM1"); }</pre>
virtual void OnInit(Msg&)	<p>Используется при инициализации объекта. В качестве параметра передается ссылка на сообщение, содержащее информацию по объекту</p>	<pre>virtual void OnInit (Msg& msg) { OnChange (msg, msg);</pre>

Метод	Описание	Пример
		<pre> }</pre>
virtual void OnChange(Msg&,Msg&)	Используется при изменении объекта. В качестве параметра передаются ссылки на сообщения, содержащие информацию по объекту до и после изменения соответственно	<pre> virtual void OnChange(Msg& msg, Msg& prev) { React react (msg.GetSourceType(), msg.GetSourceId(),"INIT"); react.SetParam("port",msg.GetPara m("port")); m_pCore->DoReact(react); }</pre>
virtual void OnDelete(Msg&)	Используется при удалении объекта. В качестве параметра передается ссылка на сообщение, содержащее информацию по объекту	<pre> virtual void OnDelete (Msg& msg) { React react (msg.GetSourceType(), msg.GetSourceId(),"EXIT"); m_pCore-> DoReact(react); }</pre>
virtual void OnEnable(Msg&)	Предназначен для обработки нажатия кнопки Disable на панели ПК <i>Интеллект</i> при включении объекта. В качестве параметра передается ссылка на сообщение, содержащая информацию по объекту	
virtual void OnDisable(Msg&)	Предназначен для обработки нажатия кнопки Disable на панели ПК <i>Интеллект</i> при выключении объекта. В качестве параметра передается ссылка на сообщение, содержащая информацию по объекту	
virtual BOOL OnEvent(Event&)	Служит для обработки событий, передаваемых в качестве параметра	<pre> virtual BOOL OnEvent(Event& event) {</pre>

Метод	Описание	Пример
		<pre> If (event.GetAction() == "ACCESS_IN" event.GetAction() == "ACCESS_OUT") { Msg per = m_pCore-> FindPersonInfoByCard(event.GetPar am("facility_code"), event.GetParam("card")); event.SetParam ("param0", ! per.GetSourceId().IsEmpty() ? per.GetParam("name") : event.GetParam("facility_code") + event.GetParam("card")); event.SetParam("param1", per.GetSourceId()); } Else If (event.GetAction() == "NOACCESS_CARD") { event.SetParam ("param0",event.GetParam("facilit y_code") + event.GetParam("card")); } return TRUE; } </pre>
virtual BOOL OnReact(React&)	Служит для обработки реакций, передаваемых в качестве параметра	

В глобальной функции `CreateNissObject(CoreInterface* core)` необходимо создать экземпляры описанных объектов, поместить их в массив `CNissObjectDLLExtArray` и вернуть указатель на объект этого массива. Через эту функцию необходимо получить указатель на интерфейс ядра, который в дальнейшем используется в объектах для обращения к методам данного интерфейса:

```

CNissObjectDLLExtArray* APIENTRY CreateNissObject(CoreInterface* core)
{
    CNissObjectDLLExtArray* ar = new CNissObjectDLLExtArray;

    ar->Add(new NissObjectDemo(core));

    ar->Add(new NissObjectDemoDevice(core));

    return ar;
}

```

После загрузки DLL-файла ядро вызывает функцию `CreateNissObject` и получает указатели на все используемые объекты.

Все панели настроек объектов хранятся в ресурсах в виде диалогов. Идентификаторы диалогов строятся по схеме **IDD_object_SETUP**, где **object** – это имя соответствующего объекта. Например, для объекта **DEMO** – это **IDD_DEMO_SETUP**, а для объекта **DEMO_DEVICE** – это **IDD_DEMO_DEVICE_SETUP**.

Примечание.

Для того чтобы в дереве настроек у объекта отображался свой собственный значок, необходимо в ресурсах DLL-файла создать **BITMAP** размером 14x14 с именем объекта.

12.2.6 Разработка RUN-файла

Управление устройствами выполняется через обмен сообщениями (командами) между RUN-файлом и ядром системы. Для реализации данного взаимодействия программного модуля с ядром используется *IIDK*, подробно рассмотренный в разделе *Intellect Integration Developer Kit (IIDK)*. Необходимую информацию можно также почерпнуть из исходных файлов демонстрационного модуля, которые прилагаются к документации.

Ниже приведен пример использования средств разработки *IIDK* для демонстрационного модуля *DEMO*.

```

CString port = "1100";

CString ip = "127.0.0.1";

CString id = "";

BOOL IsConnect = Connect (ip, port, id, myfunc);

if (!IsConnect)
{
    // не удалось подключиться
}

```



```

AfxMessageBox("Error");

Return;

}

SendMsg(id,"CAM|1|REC"); // поставить камеру 1 на запись

SendMsg(id, "DEMO|1|RESTORE"); // восстановление связи с объектом DEMO

//включить устройство DEMO_DEVICE с адресом 1

SendMsg(id,"DEMO_DEVICE|1|ON|params<1>,param0_name<address>,param0_val<1>");

Disconnect(id);

```

Внимание!

Если создан mdl-файл, то для подключения к ядру ПК *Интеллект* объект **Интерфейс IIDK** в системе не создается. В качестве идентификатора подключения передается пустая строка, то есть id равен "".

При выгрузке модуля ему посылается сообщение **WM_EXIT**:

```
#define WM_EXIT (WM_USER+2000)
```

Используя функцию WinAPI – *PostThreadMessage*, необходимо перехватить это сообщение и обеспечить корректную выгрузку модуля. В *VC++* и *MFC* сообщение **WM_EXIT** отлавливается в классе, наследуемом от *CWinApp*, в *Delphi* и *CBuilder – TApplication*.

12.2.7 Создание и настройка интегрированных объектов (модулей) в ПК Интеллект

Для создания и настройки интегрированных объектов (модулей) в ПК *Интеллект* необходимо выполнить следующие действия:

1. Разместить MDL и RUN-файлы в каталоге *Интеллект\Modules*.
2. Запустить ПК *Интеллект*.
3. На базе объекта **Компьютер** создать добавленный с помощью программного модуля объект. В результате создания станет доступна панель настройки объекта.

Примечание

Вы можете запросить у технической поддержки ITV пример проекта интеграции, в котором есть демонстрационные объекты.

4. Внести необходимые изменения на панели настроек объекта.
5. Нажать кнопку **Применить**.
6. Повторить действия пунктов 3-5 для всех нужных объектов.

В результате новые объекты будут созданы и настроены в ПК *Интеллект*.

13 Элемент управления ActiveX CamMonitor.ocx

13.1 Общее описание ActiveX-компонента CamMonitor.ocx

На странице:

- [Общее описание CamMonitor.ocx](#)
- [Требования к разработчику](#)

13.1.1 Общее описание CamMonitor.ocx


CamMonitor.ocx представляет собой ActiveX компонент, который является полным аналогом интерфейсного объекта **Монитор видеонаблюдения**. Он позволяет управлять камерами, просматривать архив и т.д.

Компонент CamMonitor.ocx поддерживает работу в демо-режиме.

13.1.2 Требования к разработчику

Для использования CamMonitor.ocx требуется:

1. знание любого языка программирования, поддерживающего использование компонентов Component Object Model (COM);
2. знание основ программирования в Win32/Win64;
3. наличие среды разработки, поддерживающей работу с ocx-файлами.

 **Требования к программному обеспечению, используемому при интеграции**

13.2 Установка CamMonitor.ocx

Внимание!

Не рекомендуется устанавливать ПК *Интеллект* и компонент CamMonitor.ocx на одном компьютере. Если это все же необходимо, то их версии должны совпадать, в противном случае работа компонента CamMonitor.ocx не гарантируется.

Установка CamMonitor.ocx осуществляется при помощи файла CamMonitorInstaller.exe, который располагается в папке <Директория установки *Интеллект*>\Redist\CamMonitor. Доступна как 32-битная, так и 64-битная версия данного компонента. Файлы установки компонентов разной битности расположены в соответствующих папках (x86 и x64 соответственно).

Если на компьютере установлен ПК *Интеллект*, то при установке файл 32-битного компонента CamMonitor.ocx помещается в папку <Директория установки ПК *Интеллект*>\Modules\, а файл 64-битного компонента устанавливается в папку <Директория установки ПК *Интеллект*>\Modules64\.

Если ПК *Интеллект* не установлен, то 32-битный компонент установится в папку c:\Program Files (x86)\ITV VideoPlayer\Modules\, а 64-битный – в папку c:\Program Files\ITV VideoPlayer x64\Modules64\

Также при установке осуществляется стандартная регистрация CamMonitor.ocx как компонента ActiveX.

CamMonitorInstaller.exe выполняет установку требуемых файлов для всех пользователей системы.

Помимо самой библиотеки устанавливается также пакет драйверов CodecPack и утилита ITV VideoPlayer, которая работает с использованием компонента CamMonitor.ocx и позволяет просматривать видеоархив по выбранной камере. Данная утилита по умолчанию устанавливается в папку C:\Program Files\ITV VideoPlayer\. Интерфейс утилиты схож с интерфейсом Converter.exe, однако в ней отсутствуют некоторые функции Converter.exe, не связанные с просмотром архива. Данную утилиту можно использовать для проверки корректности установки CamMonitor.ocx.

13.3 Параметры CamMonitor.ocx

На странице:

- [CamMenuOptions](#)
- [CamMenuProcessingOptions](#)
- [CamButtonsOptions](#)
- [MainPanelOptions](#)
- [KeysOptions](#)
- [OverlayMode](#)
- [Пример использования параметров](#)

В этом разделе перечислены параметры, позволяющие настроить режим работы компонента CamMonitor: задать отображаемые элементы интерфейса, а также режим использования оверлея.

Все параметры представляют собой целое число типа long.

Значения параметров для настройки элементов интерфейса, перечисленные в таблицах, сформированы таким образом, чтобы в двоичном представлении числа была только одна единица. Чтобы задать значение параметра, необходимо при помощи операции исключающего ИЛИ (XOR) объединить требуемые значения параметров, получив таким образом некое число, разряды которого в двоичном представлении говорят о том, какие элементы интерфейса следует отображать, а какие нет. См. также [Пример использования параметров](#).

Параметр OverlayMode отличается от прочих: он принимает значения от 0 до 2, и его значение задает режим использования оверлея.

13.3.1 CamMenuOptions

CamMenuOptions : long

Позволяет настроить функциональное меню камеры.

Допускается установка одного или нескольких флагов.

Возможные значения:

Значение	Описание
#define MENU_ENABLE_OPTION 0x00000001	Не используется. См. BUTTON_MENU_ENABLE_OPTION
#define MENU_ARM_ENABLE_OPTION 0x00000002	Отображать пункт меню Поставить на охрану
#define MENU_REC_ENABLE_OPTION 0x00000004	Отображать пункт меню Начать запись

#define MENU_CAMS_ENABLE_OPTION 0x00000008	Отображать пункт меню Камера
#define MENU_TITLES_ENABLE_OPTION 0x00000010	Отображать пункт меню Отображение титров
#define MENU_PROCESSING_ENABLE_OPTION 0x00000020	Отображать пункт меню Обработка
#define MENU_EXPORT_ENABLE_OPTION 0x00000040	Отображать пункт меню Экспорт

13.3.2 CamMenuProcessingOptions

CamMenuProcessingOptions : long

Позволяет настроить меню **Обработка** в функциональном меню камеры.

Допускается установка одного или нескольких флагов.

Возможные значения:

Значение	Описание
#define MENU_PROCESSING_DEINTERLACE_ENABLE_OPTION 0x00000001	Отображать пункт меню Деинтерлейсинг
#define MENU_PROCESSING_ZOOM_ENABLE_OPTION 0x00000002	Отображать пункт меню Увеличение . <i>Примечание. При сбросе данной опции пункт меню Обработка->Увеличение исчезает, но остаётся возможность увеличить изображение с помощью колёсика мыши.</i>
#define MENU_PROCESSING_CONTRAST_ENABLE_OPTION 0x00000004	Отображать пункт меню Контрастирование
#define MENU_PROCESSING_MASK_ENABLE_OPTION 0x00000008	Отображать пункт меню Маска детектора
#define MENU_PROCESSING_SHARP_ENABLE_OPTION 0x00000010	Отображать пункт меню Резкость

13.3.3 CamButtonsOptions

CamButtonsOptions : long

Позволяет настроить отображение кнопок компонента CamMonitor.

Допускается установка одного или нескольких флагов.

Возможные значения:

Значение	Описание
#define BUTTON_MODE_ENABLE_OPTION 0x00000100	Отображать кнопку входа в архив
#define BUTTON_TIME_ENABLE_OPTION 0x00000002	Отображать время
#define BUTTON_NAME_ENABLE_OPTION 0x00000004	Отображать название камеры
#define BUTTON_MENU_ENABLE_OPTION 0x00000008	Отображать кнопку вызова функционального меню
#define BUTTON_RAYS_ENABLE_OPTION 0x00000010	Не используется

#define BUTTON_MICS_ENABLE_OPTION 0x00000020	Не используется
#define BUTTON_ARCH_PANEL_ENABLE_OPTION 0x00000200	Отображать элементы навигации по архиву

13.3.4 MainPanelOptions

MainPanelOptions : long

Позволяет настроить отображение панели инструментов CamMonitor.

Допускается установка одного или нескольких флагов.

Возможные значения:

Значение	Описание
#define MAIN_PANEL_ENABLE_OPTION 0x00000001	Включает отображение панели
#define MAIN_PANEL_ENABLE_SCREEN_BUTTON 0x00000010	Отображать кнопку Экраны (см. Раскладка Окн видеонаблюдения на Мониторе видеонаблюдения).
#define MAIN_PANEL_ENABLE_BOOKMARK_BUTTON 0x00000020	Отображать кнопку Создать закладку (см. Создание закладок).
#define MAIN_PANEL_ENABLE_BOOKMARK_REVIEW_BUTTON 0x00000040	Отображать кнопку Созданные закладки (см. Список закладок).
#define MAIN_PANEL_ENABLE_AVIEXP_BUTTON 0x00000080	Отображать кнопку Фоновый экспорт (см. Утилита AviExport).

13.3.5 KeysOptions

KeysOptions : long

Позволяет настроить управление компонентом при помощи клавиатуры и мыши.

Допускается установка одного или нескольких флагов.

Возможные значения:

Значение	Описание
#define KEYS_ENABLE_OPTION 0x00000001	Включает возможность управления компонентом CamMonitor при помощи горячих клавиш, доступных для Монитора видеонаблюдения (см. Монитор видеонаблюдения).
#define TELEMETRY_DISABLE_OPTION 0x00000002	Отключает возможность управления телеметрией из компонента CamMonitor (см. Управление поворотными устройствами).
#define ARCH_DELETE_ENABLE_OPTION 0x00000004	Включает возможность удаления записей архива из списка видеозаписей (см. Удаление видеозаписей из архива).
#define ARCH_PROTECT_ENABLE_OPTION 0x00000008	Включает возможность защищать записи архива в списке видеозаписей (см. Защита записей и снятие защиты).

13.3.6 OverlayMode

OverlayMode : long

Задаёт режим отображения.

Возможные значения:

Значение	Описание
0	Не использовать Overlay
1	Overlay 1
2	Overlay 2

13.3.7 Пример использования параметров

```

DWORD options = CamMonitor1->CamMenuOptions;

options = options^MENU_CAMS_ENABLE_OPTION^MENU_ARM_ENABLE_OPTION^MENU_REC_ENABLE_OPTION;

CamMonitor1->CamMenuOptions = options;

CamMonitor1->CamMenuProcessingOptions ^= MENU_PROCESSING_MASK_ENABLE_OPTION;

```

13.4 Методы CamMonitor.ocx

На странице:

- [Connect](#)
- [ShowCam](#)
- [DoReactMonitor](#)
- [RemoveAllCams](#)
- [IsConnected](#)
- [GetCurlp](#)
- [SendRawMessage](#)
- [Disconnect](#)
- [SetCallBackOptions](#)
- [SetParam](#)

13.4.1 Connect

Connect(BSTR **ip**, BSTR **login**, BSTR **password**, BSTR **arch_password**, long **param**, long **port**) установка соединения с сервером/видеослужбам/долговременным архивом.

- BSTR **ip** – IP адрес сервера.
- BSTR **login** – логин для соединения с сервером (может быть пустым).
- BSTR **password** – пароль на соединение с сервером (может быть пустым).
- BSTR **arch_password** – пароль для доступа к архиву (т.е. пароль администратора, может быть пустым).
- long **param** – роль, исполняемая сервером. Параметр является обязательным.
 - 0 – видеосервер;

- 1 – оперативный архив;
- 2 – видеошлюз.
- **long port** – задает порт подключения к серверу.
 - если передать 0, 1 или 2, то соединение будет устанавливаться с портом 900, 901 или 902 соответственно;
 - если передать 100, то будет выполняться соединение через порт 10504;
 - если передать какое-либо другое значение, то будет осуществляться соединение по порту с номером port + 20000. Например, если port=900, то подключение происходит к порту сервера 20900.

Установка связи с сервером происходит **асинхронно**.

Внимание!

Если при вызове метода Connect() не указаны логин и пароль, то в элементе управления будет доступен просмотр видео со всех камер. Если разграничение прав важно в стороннем приложении, это следует учитывать на этапе разработки.

13.4.2 ShowCam

ShowCam(long **cam_id**, long **compress**, long **show**) выводит/скрывает камеру с экрана

- long **cam_id** – идентификатор(номер) камеры.
- long **compress** – уровень компрессии видео 0-5 (для локальной камеры =0). -1 - отображает в оригинальном формате транслируемом с камеры без рекомпрессии.
- long **show** – флаг означающий действие: показать/скрыть камеру (1/0).

13.4.3 DoReactMonitor

DoReactMonitor(BSTR **react_string**) – управление поведением монитора/камер

- BSTR **react_string** – строковое представление реакции.

Пример формирования react_string:

```
react_string = "MONITOR| |ARCH_FRAME_TIME|cam<3>,date<dd-mm-yy>,time<hh:mm:ss>,mode<1>";
CamMonitor1->DoReactMonitor(react_string);
```

Результат вызова функции с таким параметром: камера 3 будет переведена в режим архива видеосервера, и архив будет позиционирован на дату «dd-mm-yy» и время «hh:mm:ss» (дату и время необходимо задавать только в таком формате). Параметр mode может принимать следующие значения:

- 0 – видеошлюз, если задан (если не задан, то видеосервер).
- 1 – видеосервер.
- 2 – долговременный архив.

“MONITOR|<id игнорируется >|ARCH_FRAME_TIME|...”

Примечание:

Опционально можно указать также точность при позиционировании до миллисекунд например:
DoReactMonitor(“MONITOR||ARCH_FRAME_TIME|cam<3>,date<02-10-05>,time<12:12:22.345>,mode<1>”)

Пример. Вывод 2-го потока камеры 14 на монитор 1:

```
"MONITOR|1|ADD_CAM|cam<14>,cam_id<14>,compress<1>,stream_id<14.2>"
```

Пример. Задание частоты кадров равной 1 при просмотре архива с камеры 11

```
"MONITOR||CAM_PARAMS|cam<11>;arch_fps<1>"
```

13.4.4 RemoveAllCams

RemoveAllCams() : **long** – удаление всех камер с экрана

13.4.5 IsConnected

IsConnected() : **boolean** - метод говорит о наличии/отсутствии связи с видеосервером.

13.4.6 GetCurlp

GetCurlp() : **BSTR** – возвращает IP адрес сервера, ранее указанный при вызове **Connect**.

13.4.7 SendRawMessage

SendRawMessage(BSTR msg) – дать видеосерверу команду на исполнение.

- BSTR **msg** – строковое представление команды.

Примеры вызова функции:

```
m_Cam.SendRawMessage("CAM|1|REC");
```

```
m_Cam.SendRawMessage("CAM|1|REC_STOP");
```

```
m_Cam.SendRawMessage("CAM|1|ARM");
```

```
m_Cam.SendRawMessage("CAM|1|DISARM");
```

13.4.8 Disconnect

Disconnect() – осуществляет отсоединение от видеосервера.

13.4.9 SetCallbackOptions

SetCallbackOptions(int cam_id, int options) – задает параметры получения видеоизображения с камеры.

- int **cam_id** – идентификатор(номер) камеры.
- int **options** – опции. Возможные значения параметра **options**:
 - WithoutVideoFrame = 0x00 – не присылать кадры из модуля видео.
 - WithVideoFrame = 0x01 – присылать кадры из модуля видео.
 - WithExtendedParams = 0x02 – получать видео кадры с дополнительными параметрами (время, число кадров, субтитры).
 - WithInformationLayout = 0x04 – отображать видео в окне с элементами управления (контекстное меню).
 - WithCompressedData = 0x08 – отображать видео в исходном формате без распаковки (если оно есть).
 - WithoutDecode = 0x10 – отключить декодирование видео на сервере.
 - WithoutSubtitles=0x20 – отключить субтитры.

Примечание.

Параметр options формируется так же, как параметры компонента CamMonitor.ocx – см. [Параметры CamMonitor.ocx](#)

13.4.10 SetParam

SetParam(BSTR **param_name**, BSTR **param_value**) - задает количество окон видеокамер в CamMonitor.

- BSTR **param_name** - строковое представление длины или ширины.
- BSTR **param_value** - количество окон видеокамер.

Примеры вызова функции:

```
m_cam.SetParam("monitor_ch", m_NH);
m_cam.SetParam("monitor_cw", m_NW);
```

13.5 События CamMonitor.ocx

OnCamListChange (long **cam_id**, long **action**) - возникает при установлении связи с сервером или при изменении количества камер на сервере

- long **cam_id** – идентификатор камеры.
- long **action** – равен 1, если камера с **id == cam_id** существует, иначе **action == 0**.

Данное событие возникает столько раз, сколько камер на данном видеосервере. Признаком окончания вызовов **OnCamListChange** является отрицательное значение параметра **cam_id** (**cam_id < 0**).

Если на сервере находится, например, 3 камеры (1, 2, 3), то последовательно возникнут события:

CamListChange(1,1)

CamListChange(2,1)

CamListChange(3,1)

CamListChange(-1,1)

Пример:

Показать камеру с **cam_id = 2** с уровнем компрессии **compress = 1**;

```
CamMonitor1CamListChange(long cam_id, long action)
{
    if(cam_id == -1)
    {
        CamMonitor1->ShowCam(2,1,1);
    }
}
```

14 HTTP API ПК Интеллект

14.1 Общие сведения о HTTP API

Содержание

- [Программная реализация](#)
- [Возможности](#)
- [Особенности и ограничения](#)
- [Авторизация](#)
- [Формат ответа по умолчанию](#)
- [Кросс-доменные запросы \(CORS\)](#)
- [Использование HTTPS](#)

14.1.1 Программная реализация

Программно HTTP API предоставляется модулем **web2 – Веб-сервер 2.0** (см. [Настройка Сервера для подключения Клиентов с помощью модуля Веб-сервер 2.0](#)).

14.1.2 Возможности

HTTP API позволяет использовать следующие функции:

1. Получать сведения об интерактивных картах: список карт, имя карты, список слоев карты, параметры слоя, фоновый рисунок слоя, информация о списке точек и отдельной точке на слое (см. [Карта](#)).
2. Получать сведения о классах объектов, созданных на Сервере, список состояний для класса объектов и информацию о состоянии, а также иконки для определенного состояния (см. [Классы объектов](#)).
3. Получать список объектов, созданных на сервере, информацию об отдельном объекте, состоянии объекта, список доступных действий с объектом (см. [Объекты](#)).
4. Получать события с Сервера как отдельно, так и блоками (см. [Получение событий](#)).
5. Отсылать команды на Сервер (см. [Отсылка команд на сервер](#)).
6. Запускать выполнение макрокоманд (см. [Макрокоманды](#)).
7. Работать с видео: получить кадры, запрашивать конфигурацию, получать живое видео и архив, управлять записью, ставить и снимать камеры с охраны, управлять телеметрией (см. [Видео](#)).
8. Получать живой и архивный звук (см. [Звук](#)).
9. Получать список пользователей (см. [Пользователи](#)).
10. Отправлять события и реакции в ядро ПК *Интеллект* (см. [Отправка реакций и событий в ПК Интеллект по HTTP-запросу](#)).
11. Обращаться к API Face-Интеллект и API Авто-Интеллект (см. [Обращение к API вертикальных решений через HTTP API ПК Интеллект](#)).
12. Настраивать интеграции с Техносерв и ЕЦХД (см. [Настройка интеграции с Техносерв и Команды, используемые для интеграции ЕЦХД](#)).

14.1.3 Особенности и ограничения

1. В примерах, приводимых в данном разделе, используются следующие обозначения:
 - а. **Port** – порт. По умолчанию порт модуля **Веб-сервер 2.0** – 8085. Указывать порт при отправке команд HTTP API **обязательно**.
 - б. **/web2** – веб-контекст, в котором работает модуль web2. Это контекст веб-приложения. Далее описание будет опускаться там, где действие запроса понятно из контекста.
2. URL, id объектов и расширения файлов чувствительны к регистру.
3. Дата и время указываются в формате RFC3339, подробнее см. <http://www.ietf.org/rfc/rfc3339.txt>.
4. Должно быть отключено сжатие сообщений от ядер, то есть ключ реестра MsgCompressOn=0 (см. [Справочник ключей для Интеллекта базового](#)).

14.1.4 Авторизация

Для выполнения запросов необходима авторизация. Поддерживаются 2 типа авторизации: Basic и Bearer.

При авторизации Basic данные пользователя необходимо добавлять во все HTTP запросы в следующем виде:

```
http://[имя_пользователя]:[пароль]@[IP-адрес]:[порт]/web2
```

При авторизации Bearer используется токен, полученный от web-сервера (см. [Авторизация в ПК Интеллект по token ключу](#)).

14.1.5 Формат ответа по умолчанию

По умолчанию формат ответа JSON. Включение ответа по умолчанию в формате XML осуществляется на панели настройки объекта **Веб-сервер 2.0** (см. [Настройка типа ответа на запросы HTTP API по умолчанию](#)). Также формат ответа может быть явно указан в заголовке **Accept**, например **application/json** или **application/xml**.

Указанный формат ответа в запросе имеет больший приоритет, чем заданный на панели настройки объекта **Веб-сервер 2.0** формат ответа по умолчанию.


14.1.6 Кросс-доменные запросы (CORS)

Для выполнения кросс-доменных запросов или для получения доступа к необходимым заголовкам в ответе (например, в связи с ограничениями из-за политики CORS браузера), необходимо в заголовке запроса указать **Origin** (домен сайта, с которого происходит запрос). В таком случае в ответе будет содержаться заголовок **Access-Control-Allow-Origin**, который указывает на разрешение доступа к ресурсу с указанного домена кросс-сайтовым способом. Заголовок **Access-Control-Allow-Origin: *** означает, что к ресурсу можно получить доступ с любого домена кросс-сайтовым способом.

14.1.7 Использование HTTPS

По умолчанию Веб-сервер 2.0 использует протокол HTTP. Чтобы передавать данные по протоколу HTTPS, его использование нужно включить в **Веб-сервере 2.0** и в объекте **Web-сервер** (см. [Настройки HTTPS](#) и [Задание параметров подключения Клиентов к Web-серверу](#)). Порт для HTTPS по умолчанию – 8443.

Если использование HTTPS в настройках не включено, но в запросах при этом используется порт 8443, то будет происходить переадресация на HTTP. Например, при запросе «https://127.0.0.1:8443/web2/secure/configuration» автоматически произойдет переход на «http://127.0.0.1:8085/web2/secure/configuration».

 [Обращение к API вертикальных решений через HTTP API ПК Интеллект](#)

14.2 Версия продукта

14.2.1 Общий формат запроса:

```
GET http://IP-адрес:порт/web2/product/version
```

14.2.2 Пример запроса:

```
GET http://127.0.0.1:8085/web2/product/version
```

14.2.3 Пример ответа:

В ответ приходит text/plain строка вида:

```
Intellect/4.11.2.2875
```

Это означает, что сервер поддерживает протокол, описанный в данном документе. Строка может меняться в зависимости от версии продукта. Это позволяет различать два схожих по функциональным возможностям, но разных по протоколу веб-сервера в разных продуктах.

14.3 Авторизация в ПК Интеллект по token ключу

Авторизация в ПК *Интеллект* по token ключу позволяет:

- В url-запросе вместо указания параметров "login" и "password" указывать токен в параметре "token".
Пример запроса видео с авторизацией в ПК *Интеллект* по token ключу:

```
http://127.0.0.1:80/video/action.do?
normalize=true&version=4.10.0.0&video_in=CAM:1&token=EoHWC_zXFILImB0hL4QgjPc5624cJXMF
```

- Использовать в заголовке запроса в параметре "Authorization" как Bearer Token Authentication.
Например:

```
Authorization: Bearer "PJ_eHSwUsqjXX7PRZMB8hm_zKEncg3hE"
```

Токен может получить только зарегистрированный пользователь ПК *Интеллект*. Если в запросе не указать логин и пароль для Basic авторизации, то сервер вернёт ошибку «500 Internal Server Error».

14.3.1 Общий формат запроса:

GET/POST http://{login}:{password}@IP-адрес:порт/token?expires_in={expires_in}

14.3.2 Параметры запроса:

Параметр	Обязательный	Описание
login	Да	Имя пользователя ПК <i>Интеллект</i>
password	Да	Пароль пользователя ПК <i>Интеллект</i>
expires_in	Нет	<p>Время действия токена в секундах. Максимальное значение – 1 день. Токен перестает действовать через указанный период времени.</p> <p>Значение по умолчанию: 1800.</p> <p>Для того, чтобы разлогиниться, указать значение 0.</p> <p><i>Примечание. Для одного пользователя может существовать только 1 токен.</i></p>

14.3.3 Пример запроса:

GET/POST http://USER:PASSWORD@127.0.0.1:80/token?expires_in=1800

14.3.4 Пример ответа:

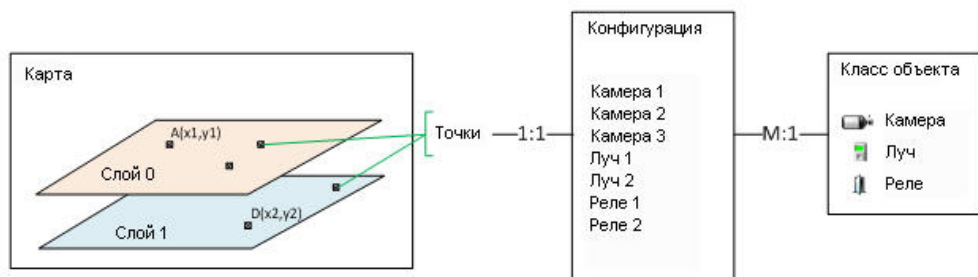
```
{
  "access_token": "PJ_eHSwUsqjXX7PRZMB8hm_zKEncg3hE",
```

```
"token_type":"bearer",
"expires_in":"1800"
}
```

14.3.5 Параметры ответа:

Параметр	Описание
access_token	Токен
token_type	Тип токена
expires_in	Время действия токена в секундах

14.4 Карта



На Сервере может быть создано несколько карт. Каждая карта может содержать один и более слоёв. На каждом слое расположены точки. Каждая точка связана с одним из объектов конфигурации.

Конфигурация – это объекты ПК *Интеллект*. Каждый объект является объектом определённого класса. Каждый объект имеет одно состояние и список действий, которые можно с ним производить.

Класс объекта описывает его вид (значки), возможные состояния и возможные действия с объектом в каждом из состояний.

14.4.1 Получение списка карт

14.4.1.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/kartas/

14.4.1.2 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/kartas/

14.4.1.3 Пример ответа:

```
<kartas>
  <karta>
    <id>2</id>
    <layers>
```

```

<layer>
  <geo_h>10.0</geo_h>
  <height>578</height>
  <id>2</id>
  <lat_bl>43.5905</lat_bl>
  <lat_br>43.6875</lat_br>
  <lat_c>43.6395</lat_c>
  <lat_tl>43.5914</lat_tl>
  <lat_tr>43.6885</lat_tr>
  <lon_bl>43.4584</lon_bl>
  <lon_br>43.4599</lon_br>
  <lon_c>43.4809</lon_c>
  <lon_tl>43.5018</lon_tl>
  <lon_tr>43.5033</lon_tr>
  <mapId>2</mapId>
  <name>Layer 2</name>
  <points>
    <point>
      <id>CAM:1</id>
      <layerId>2</layerId>
      <mapId>2</mapId>
      <angle>0.0</angle>
      <geo_angle>0.0</geo_angle>
      <latitude>43.47727</latitude>
      <longitude>43.602381</longitude>
      <x>95.0</x>
      <y>329.0</y>
    </point>
  </points>
  <width>800</width>
  <zoomDef>1.0</zoomDef>
  <zoomMax>4.0</zoomMax>
  <zoomMin>0.25</zoomMin>
  <zoomStep>0.25</zoomStep>
</layer>
</layers>
<name>Map 2</name>
</karta>
</kartas>

```

14.4.1.4 Параметры ответа

Параметр	Описание
Параметры группы <karta>	
id	Идентификатор карты
name	Название карты
layers	Список слоев
Параметры группы <layer>	

Параметр	Описание
geo_h	Метка высоты (см. Настройка привязки карты к координатной сетке)
height	Высота слоя карты в пикселях
width	Ширина слоя карты в пикселях
id	Идентификатор слоя
lat_bl	Широта: нижний левый угол
lat_br	Широта: нижний правый угол
lat_c	Широта: центр
lat_tl	Широта: верхний левый угол
lat_tr	Широта: верхний правый угол
lon_bl	Долгота: нижний левый угол
lon_br	Долгота: нижний правый угол
lon_c	Долгота: центр
lon_tl	Долгота: верхний левый угол
lon_tr	Долгота: верхний правый угол
mapId	Идентификатор карты
name	Название слоя
points	Список точек на слое
zoomDef	Масштаб по умолчанию
zoomMax	Минимальный масштаб
zoomMin	Максимальный масштаб
zoomStep	Шаг масштабирования
Параметры группы <point>	
id	Тип и идентификатор объекта в формате ТИП:ID
layerId	Идентификатор слоя

Параметр	Описание
mapId	Идентификатор карты
angle	Угол поворота значка объекта
geo_angle	Угол обзора (для камеры, см. Настройка отображения угла обзора камеры на Карте)
latitude	Широта (координата точки) Параметр имеет ненулевое значение в следующих случаях: <ol style="list-style-type: none"> 1. В качестве подложки слоя используется внешняя карта (см. Настройка сервера внешних карт) 2. Настроена геопривязка слоя (см. Настройка привязки карты к координатной сетке)
longitude	Долгота (координата точки) Параметр имеет ненулевое значение в следующих случаях: <ol style="list-style-type: none"> 1. В качестве подложки слоя используется внешняя карта (см. Настройка сервера внешних карт) 2. Настроена геопривязка слоя (см. Настройка привязки карты к координатной сетке)
x	Координата точки относительно подложки по оси X
y	Координата точки относительно подложки по оси Y

14.4.2 Информация об одной карте

14.4.2.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/kartas/{plan}/

14.4.2.2 Параметры запроса:

Параметр	Обязательный	Описание
plan	Да	Идентификатор карты

14.4.2.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/kartas/2

14.4.2.4 Пример ответа:

```
<karta>
  <id>2</id>
  <name>This is plan of a building</name>
</karta>
```


14.4.2.5 Параметры ответа:

Параметр	Описание
id	Идентификатор карты
name	Название карты

14.4.3 Список слоёв для выбранной карты

14.4.3.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/kartas/{plan]/layers

14.4.3.2 Параметры запроса

Параметр	Обязательный	Описание
plan	Да	Идентификатор карты

14.4.3.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/kartas/2/layers

14.4.3.4 Пример ответа:

```
<layers>
  <layer>
    <geo_h>10.0</geo_h>
    <height>578</height>
    <id>2</id>
    <lat_bl>43.5905</lat_bl>
    <lat_br>43.6875</lat_br>
    <lat_c>43.6395</lat_c>
    <lat_tl>43.5914</lat_tl>
    <lat_tr>43.6885</lat_tr>
    <lon_bl>43.4584</lon_bl>
    <lon_br>43.4599</lon_br>
    <lon_c>43.4809</lon_c>
    <lon_tl>43.5018</lon_tl>
    <lon_tr>43.5033</lon_tr>
    <mapId>2</mapId>
    <name>Layer 2</name>
    <points>
      <point>
        <id>CAM:1</id>
        <layerId>2</layerId>
        <mapId>2</mapId>
        <angle>0.0</angle>
        <geo_angle>0.0</geo_angle>
        <latitude>43.47727</latitude>
```

```

        <longitude>43.602381</longitude>
        <x>95.0</x>
        <y>329.0</y>
    </point>
</points>
<width>800</width>
<zoomDef>1.0</zoomDef>
<zoomMax>4.0</zoomMax>
<zoomMin>0.25</zoomMin>
<zoomStep>0.25</zoomStep>
</layer>
</layers>

```

14.4.3.5 Параметры ответа

Параметр	Описание
Параметры группы <layer>	
geo_h	Метка высоты (см. Настройка привязки карты к координатной сетке)
height	Высота слоя карты в пикселях
width	Ширина слоя карты в пикселях
id	Идентификатор слоя
lat_bl	Широта: нижний левый угол
lat_br	Широта: нижний правый угол
lat_c	Широта: центр
lat_tl	Широта: верхний левый угол
lat_tr	Широта: верхний правый угол
lon_bl	Долгота: нижний левый угол
lon_br	Долгота: нижний правый угол
lon_c	Долгота: центр
lon_tl	Долгота: верхний левый угол
lon_tr	Долгота: верхний правый угол
mapId	Идентификатор карты
name	Название слоя

Параметр	Описание
points	Список точек на слое
zoomDef	Масштаб по умолчанию
zoomMax	Минимальный масштаб
zoomMin	Максимальный масштаб
zoomStep	Шаг масштабирования
Параметры группы <point>	
id	Тип и идентификатор объекта в формате ТИП:ID
layerId	Идентификатор слоя
mapId	Идентификатор карты
angle	Угол поворота значка объекта
geo_angle	Угол обзора (для камеры, см. Настройка отображения угла обзора камеры на Карте)
latitude	Широта (координата точки) Параметр имеет ненулевое значение в следующих случаях: <ul style="list-style-type: none"> 1. В качестве подложки слоя используется внешняя карта (см. Настройка сервера внешних карт) 2. Настроена геопривязка слоя (см. Настройка привязки карты к координатной сетке)
longitude	Долгота (координата точки) Параметр имеет ненулевое значение в следующих случаях: <ul style="list-style-type: none"> 1. В качестве подложки слоя используется внешняя карта (см. Настройка сервера внешних карт) 2. Настроена геопривязка слоя (см. Настройка привязки карты к координатной сетке)
x	Координата точки относительно подложки по оси X
y	Координата точки относительно подложки по оси Y

14.4.4 Информация о конкретном слое

14.4.4.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/kartas/{plan}/layers/{base}/

14.4.4.2 Параметры запроса:

Параметр	Обязательный	Описание
plan	Да	Идентификатор карты
base	Да	Идентификатор слоя

14.4.4.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/kartas/2/layers/2/

14.4.4.4 Пример ответа:

```
<layer>
  <height>1000</height>
  <id>2</id>
  <mapId>2</mapId>
  <name>Base layer for plan</name>
  <width>1000</width>
  <zoomDef>1.0</zoomDef>
  <zoomMax>4.0</zoomMax>
  <zoomMin>0.25</zoomMin>
  <zoomStep>0.25</zoomStep>
</layer>
```

14.4.4.5 Параметры ответа:

Параметр	Описание
height	Высота слоя карты в пикселях
width	Ширина слоя карты в пикселях
id	Идентификатор слоя
mapId	Идентификатор карты
name	Название слоя
zoomDef	Масштаб по умолчанию
zoomMax	Минимальный масштаб
zoomMin	Максимальный масштаб
zoomStep	Шаг масштабирования

14.4.5 Фонový рисунок слоя

14.4.5.1 Общий формат запроса

GET http://IP-адрес:порт/web2/secure/kartas/{plan}/layers/{base}/image.{ext}

14.4.5.2 Параметры запроса

Параметр	Обязательный	Описание
plan	Да	Идентификатор карты
base	Да	Идентификатор слоя
ext	Да	Расширение файла. Допустимые значения: png или jpg

14.4.5.3 Пример запроса

GET http://127.0.0.1:8085/web2/secure/kartas/2/layers/3/image.png

14.4.5.4 Пример ответа

В ответ приходит изображение в указанном формате.

14.4.5.5 Ошибки выполнения запроса

Ошибка	Описание
404	Запрошено изображение в формате JPEG

14.4.6 Список точек на слое

14.4.6.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/kartas/{plan}/layers/{base}/points/

14.4.6.2 Параметры запроса:

Параметр	Обязательный	Описание
plan	Нет	Идентификатор карты
base	Нет	Идентификатор слоя

14.4.6.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/kartas/2/layers/2/points/

14.4.6.4 Пример ответа:

Примечание.

Если тип отображения добавленного объекта на карте отличен от **Изображение**, то на запрос приходит пустой ответ. См. также [Прикрепление объектов к слою интерактивной карты](#).

```
<points>
  <point>
    <id>CAM:1</id>
    <layerId>2</layerId>
    <mapId>2</mapId>
    <angle>0.0</angle>
    <geo_angle>0.0</geo_angle>
    <latitude>43.47727</latitude>
    <longitude>43.602381</longitude>
    <x>95.0</x>
    <y>329.0</y>
  </point>
</points>
```

14.4.6.5 Параметры ответа:

Параметр	Описание
id	Идентификатор объекта в формате "ТИП:ИДЕНТИФИКАТОР", например, "CAM:1"
layerId	Идентификатор слоя
mapId	Идентификатор карты
angle	Угол поворота значка объекта
geo_angle	Угол обзора (для камеры, см. Настройка отображения угла обзора камеры на Карте)
latitude	Широта (координата точки). Параметр имеет ненулевое значение в следующих случаях: <ol style="list-style-type: none"> 1. В качестве подложки слоя используется внешняя карта (см. Настройка сервера внешних карт) 2. Настроена геопривязка слоя (см. Настройка привязки карты к координатной сетке)
longitude	Долгота (координата точки). Параметр имеет ненулевое значение в следующих случаях: <ol style="list-style-type: none"> 1. В качестве подложки слоя используется внешняя карта (см. Настройка сервера внешних карт) 2. Настроена геопривязка слоя (см. Настройка привязки карты к координатной сетке)
x	Координата X верхнего левого угла значка объекта

Параметр	Описание
y	Координата Y верхнего левого угла значка объекта

Координатная сетка привязана к слою следующим образом:



Т.е. x и y не могут быть отрицательными, но могут быть дробными.

14.4.7 Информация об отдельной точке на слое

14.4.7.1 Общий формат запроса:

GET <http://IP-адрес:порт/web2/secure/kartas/{plan}/layers/{base}/points/{CAM:id}>

14.4.7.2 Параметры запроса:

Параметр	Обязательный	Описание
plan	Нет	Идентификатор карты
base	Нет	Идентификатор слоя
CAM:id	Да	Идентификатор объекта в формате "ТИП:ИДЕНТИФИКАТОР", например, "CAM:1"

14.4.7.3 Пример запроса:

GET <http://127.0.0.1:8085/web2/secure/kartas/plan/layers/base/points/CAM:2>

14.4.7.4 Пример ответа:

```
<point>
  <id>CAM:2</id>
  <layerId>base</layerId>
  <mapId>plan</mapId>
  <x>200.0</x>
  <y>200.0</y>
</point>
```

14.4.7.5 Параметры ответа:

Параметр	Описание
id	Идентификатор объекта в формате "ТИП:ИДЕНТИФИКАТОР", например, "CAM:1"
layerId	Идентификатор слоя
mapId	Идентификатор карты
x	Координата X верхнего левого угла значка объекта
y	Координата Y верхнего левого угла значка объекта

14.5 Классы объектов

14.5.1 Список классов объектов, которые существуют на сервере

14.5.1.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/objectClasses

14.5.1.2 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/objectClasses

14.5.1.3 Пример ответа:

```
<objectClasses>
  <objectClass>
    <id>GRELE</id>
  </objectClass>
  <objectClass>
    <id>USERS</id>
  </objectClass>
  <objectClass>
    <id>CAM</id>
  </objectClass>
  <objectClass>
    <id>RIGHTS</id>
  </objectClass>
  <objectClass>
    <id>GRAY</id>
  </objectClass>
</objectClasses>
```


14.5.1.4 Параметры ответа:

Параметр	Описание
id	Идентификатор класса объектов

14.5.2 Отдельный класс объектов

14.5.2.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/objectClasses/{objectClass}/

14.5.2.2 Параметры запроса:

Параметр	Обязательный	Описание
objectClass	Да	Идентификатор класса объектов

14.5.2.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/objectClasses/GRELE/

14.5.2.4 Пример ответа:

```
<objectClass>
  <id>GRELE</id>
</objectClass>
```

14.5.2.5 Параметры ответа:

Параметр	Описание
id	Идентификатор класса объектов

14.5.3 Список состояний для определённого класса объектов

14.5.3.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/objectClasses/{objectClass}/states/

14.5.3.2 Параметры запроса:

Параметр	Обязательный	Описание
objectClass	Да	Идентификатор класса объектов

14.5.3.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/objectClasses/GRELE/states/

14.5.3.4 Пример ответа:

```
<states>
  <state>
    <id>off</id>
  </state>
  <state>
    <id>on</id>
  </state>
  <state>
    <id>disabled</id>
  </state>
</states>
```

14.5.3.5 Параметры ответа:

Параметр	Описание
id	Идентификатор всех возможных состояний класса объектов

14.5.4 Информация о конкретном состоянии

14.5.4.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/objectClasses/{ObjectClass}/states/{State}/

14.5.4.2 Параметры запроса:

Параметр	Обязательный	Описание
ObjectClass	Да	Идентификатор класса объектов
State	Да	Идентификатор состояния класса объектов

14.5.4.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/objectClasses/GRELE/states/off/

14.5.4.4 Пример ответа:

```
<state>
  <id>off</id>
</state>
```

14.5.4.5 Параметры ответа:

Параметр	Описание
id	Идентификатор состояния класса объектов

14.5.5 Получение иконки для определённого состояния

14.5.5.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/objectClasses/{ObjectClass}/states/{State}/image.png

14.5.5.2 Параметры запроса:

Параметр	Обязательный	Описание
ObjectClass	Да	Название класса объектов
State	Да	Идентификатор состояния класса объектов

14.5.5.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/objectClasses/GRELE/states/off/image.png

14.5.5.4 Пример ответа:

В ответ придет изображение в формате png.

14.5.6 Список событий для определенного класса объектов

14.5.6.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/objectClasses/{ObjectClass}/events/

14.5.6.2 Параметры запроса:

Параметр	Обязательный	Описание
ObjectClass	Да	Идентификатор класса объектов

14.5.6.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/objectClasses/GRELE/events/

14.5.6.4 Пример ответа:

```
<events>
  <event>
```

```

    <id>23</id>
    <sid>grele.disable</sid>
    <description>Disable rele</description>
  </event>
  <event>
    <id>24</id>
    <sid>grele.enable</sid>
    <description>Enable rele</description>
  </event>
</events>

```

14.5.6.5 Параметры ответа:

Параметр	Описание
id	Идентификатор объекта
sid	Команда события
description	Описание события

14.6 Объекты

14.6.1 Получение списка всех объектов сервера

14.6.1.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/configuration?pageItems={pageItems}&page={page}

14.6.1.2 Параметры запроса:

Параметр	Обязательный	Описание
pageItems	Нет	Задаёт количество объектов, выводимых на странице. Значение должно быть больше 0, по умолчанию page=1000. Параметр page применяется только вместе с указанием параметра pageItems
page	Нет	Задаёт номер страницы, отображаемой в результате запроса. Значение должно быть больше 0, по умолчанию pageItems=1

Внимание!

Если в системе много объектов (>1000), то необходимо использовать постраничный вывод.

Обработка всех объектов производится перебором страниц до получения пустого массива.

14.6.1.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/configuration

14.6.1.4 Пример ответа:

Запрос возвращает список следующих объектов с указанием состояния:

- список камер, добавленных в Web-сервер, с указанием идентификаторов соответствующих микрофонов, динамиков, поворотного устройства, пресетов, а также мониторов и экранов, на которых отображаются камеры (см. также [Выбор и настройка видеокамер для Web-сервера](#));
- список экранов и мониторов, на которых отображаются камеры, добавленные в Web-сервер;
- список камер, добавленных на карты, выбранные для использования *Веб-сервером 2.0* (см. [Выбор карт](#));
- список лучей;
- список реле;
- список макрокоманд;
- список RTSP-серверов с указанием используемых портов и добавленных в них камер;
- список областей и разделов.

JSON:

```
[
  {
    "id": "1",
    "name": "Area 1",
    "regions": [
      {
        "id": "1.1",
        "zoneId": "1",
        "name": "Region 1.1",
        "zoneDescription": "Описание зоны"
      }
    ]
  },
  {
    "type": "DISPLAY",
    "id": "DISPLAY:1",
    "extId": "1",
    "name": "Display 1",
    "displayId": "1",
    "state": {
      "id": "normal",
      "type": "NORMAL",
      "fullState": null
    }
  },
  {
    "type": "MACRO",
    "id": "MACRO:2",
    "extId": "2",
    "name": "Macro 2",
    "state": {
      "id": "normal",
      "type": "NORMAL",
      "fullState": null
    }
  },
  {
    "type": "STREAMING_SERVER",
    "id": "STREAMING_SERVER:1",
    "extId": "1",
```

```

    "name": "RTSP Server 1",
    "state": {
      "id": "normal",
      "type": "NORMAL",
      "fullState": null
    },
    "port": "554",
    "cams": "1;;;;;"
  },
  {
    "type": "CAM",
    "id": "CAM:1",
    "extId": "1",
    "name": "Camera 1",
    "displayId": "1",
    "monitorId": "1",
    "state": {
      "id": "connected_recording",
      "type": "NORMAL",
      "fullState": "DISARMED|RECORDER_ON|RECORDING"
    },
    "presets": []
  },
  {
    "type": "SLAVE",
    "id": "SLAVE:COMP",
    "extId": "COMP",
    "name": "Computer DESKTOP-JHRURJJ",
    "state": {
      "id": "connected",
      "type": "NORMAL",
      "fullState": null
    }
  },
  {
    "type": "MONITOR",
    "id": "MONITOR:1",
    "extId": "1",
    "name": "Monitor 1",
    "displayId": "1",
    "monitorId": "1",
    "state": {
      "id": "normal",
      "type": "NORMAL",
      "fullState": null
    },
    "camList": [
      "1"
    ]
  }
]

```

XML:

```

<baseObjects>
  <CAM>

```

```

<displayId>1</displayId>
<extId>9</extId>
<geo_angle>0.0</geo_angle>
<id>CAM:9</id>
<latitude>-98.0533</latitude>
<longitude>56.4089</longitude>
<monitorId>1;2</monitorId>
<name>Camera 9</name>
<regionId />
<state>
  <fullState>DISARMED</fullState>
  <id>connected</id>
  <type>NORMAL</type>
</state>
<type>CAM</type>
<additionalInfo />
<micId />
<presets />
<speakerId />
<telemetryId />
</CAM>
<baseObject>
  <displayId>1</displayId>
  <extId>1</extId>
  <id>DISPLAY:1</id>
  <name>Display 1</name>
  <state>
    <id>normal</id>
    <type>NORMAL</type>
  </state>
  <type>DISPLAY</type>
</baseObject>
<CAM>
  <displayId>1</displayId>
  <extId>5</extId>
  <geo_angle>0.0</geo_angle>
  <id>CAM:5</id>
  <latitude>-97.6674</latitude>
  <longitude>56.3588</longitude>
  <monitorId>1</monitorId>
  <name>pos</name>
  <regionId />
  <state>
    <fullState>DISARMED</fullState>
    <id>connected</id>
    <type>NORMAL</type>
  </state>
  <type>CAM</type>
  <additionalInfo />
  <micId />
  <presets />
  <speakerId />
  <telemetryId />
</CAM>
<STREAMING_SERVER>
  <cams>1;2;3;4;5;6;9;;;;;;</cams>
  <extId>1</extId>

```

```

<id>STREAMING_SERVER:1</id>
<name>RTSP Server 1</name>
<state>
  <id>normal</id>
  <type>NORMAL</type>
</state>
<type>STREAMING_SERVER</type>
<port>5543</port>
</STREAMING_SERVER>
<CAM>
  <displayId>1</displayId>
  <extId>1</extId>
  <geo_angle>0.0</geo_angle>
  <id>CAM:1</id>
  <latitude>-98.3293</latitude>
  <longitude>56.741</longitude>
  <monitorId>1;2</monitorId>
  <name>Camera 1</name>
  <regionDescription />
  <regionId>1.1</regionId>
  <regionName>Region 1.1</regionName>
  <state>
    <fullState>DISARMED</fullState>
    <id>connected</id>
    <type>NORMAL</type>
  </state>
  <type>CAM</type>
  <zoneId>1</zoneId>
  <zoneName>Area 1</zoneName>
  <additionalInfo />
  <micId />
  <presets />
  <speakerId />
  <telemetryId />
</CAM>
<SLAVE>
  <extId>COMP</extId>
  <id>SLAVE:COMP</id>
  <name>LOCALHOST</name>
  <state>
    <id>connected</id>
    <type>NORMAL</type>
  </state>
  <type>SLAVE</type>
</SLAVE>
<baseObject>
  <displayId>1</displayId>
  <extId>2</extId>
  <id>MONITOR:2</id>
  <monitorId>2</monitorId>
  <name>Monitor 2</name>
  <state>
    <id>normal</id>
    <type>NORMAL</type>
  </state>
  <type>MONITOR</type>
</baseObject>

```



```

<baseObject>
  <displayId>1</displayId>
  <extId>1</extId>
  <id>MONITOR:1</id>
  <monitorId>1</monitorId>
  <name>Monitor 1</name>
  <state>
    <id>normal</id>
    <type>NORMAL</type>
  </state>
  <type>MONITOR</type>
</baseObject>
</baseObjects>

```

14.6.1.5 Параметры ответа:

Параметр	Описание
Общие параметры	
extId	Идентификатор объекта
id	Тип и идентификатор объекта в формате ТИП:ID
name	Название объекта
state	Состояние объекта. Описание параметров <id>, <type> и <fullState> см. в разделе Состояние отдельного объекта
type	Тип объекта
Особые параметры	
cams	Список камер в RTSP-сервере через точку с запятой
port	Порт, используемый RTSP-сервером
regionDescription	Описание области
regionId	Идентификатор раздела, в который добавлен объект
regionName	Название раздела
zonelId	Идентификатор области, в которую добавлен объект
zoneName	Название области
latitude	Широта (координата объекта). Параметр имеет ненулевое значение в следующих случаях:

Параметр	Описание
	<ol style="list-style-type: none"> 1. В качестве подложки слоя используется внешняя карта (см. Настройка сервера внешних карт) 2. Настроена геопривязка слоя (см. Настройка привязки карты к координатной сетке)
longitude	<p>Долгота (координата объекта).</p> <p>Параметр имеет ненулевое значение в следующих случаях:</p> <ol style="list-style-type: none"> 1. В качестве подложки слоя используется внешняя карта (см. Настройка сервера внешних карт) 2. Настроена геопривязка слоя (см. Настройка привязки карты к координатной сетке)
monitorId	Для камеры: идентификатор монитора, на который добавлена камера. Если идентификаторов несколько, они разделяются точкой с запятой
monitorName	Для монитора: название монитора
geo_angle	Для камеры: угол обзора (см. Настройка отображения угла обзора камеры на Карте)
additionalInfo	Для камеры: значение поля Дополнительная информация
micId	Для камеры: идентификатор связанного микрофона
presets	Для камеры: список предустановок
speakerId	Для камеры: идентификатор связанного динамика
telemetryId	Для камеры: идентификатор устройства управления телеметрией
displayId	Для интерфейсных объектов: идентификатор экрана. Если идентификаторов несколько, они разделяются точкой с запятой
displayName	Для экрана: название экрана
camList	Для монитора: список идентификаторов камер, добавленных в монитор, разделенные точкой с запятой

14.6.2 Информация об отдельном объекте

14.6.2.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/configuration/{objectClass}:{id}/

14.6.2.2 Параметры запроса:

Параметр	Обязательный	Описание
objectClass	Да	Идентификатор класса объектов
id	Да	Идентификатор объекта

14.6.2.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/configuration/GRAY:2/

14.6.2.4 Пример ответа:

```
<GRAY>
  <id>GRAY:2</id>
  <name>Луч 2</name>
  <state>
    <id>aLarmed</id>
  </state>
</GRAY>
```

14.6.2.5 Параметры ответа:

Параметр	Описание
id	Идентификатор объекта в формате "ТИП:ИДЕНТИФИКАТОР", например, "CAM:1"
name	Название объекта в ПК <i>Интеллект</i>
state id	Текущее состояние объекта

14.6.3 Состояние отдельного объекта

14.6.3.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/configuration/{objectClass}:{id}/state/

14.6.3.2 Параметры запроса:

Параметр	Обязательный	Описание
objectClass	Да	Идентификатор класса объектов
id	Да	Идентификатор объекта

14.6.3.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/configuration/GRAY:2/state/

14.6.3.4 Пример ответа:

```
<state>
  <fullState>ON, ARMED</fullState>
  <id>armed</id>
  <type>NORMAL</type>
```

</state>

14.6.3.5 Параметры ответа:

Параметр	Описание
fullState	Полное состояние объекта, хранящееся в базе данных
id	Состояние объекта в терминах HTTP API
type	Состояние объекта в терминах HTTP API

Возможные значения данных параметра fullState для Луча:

Состояние луча	fullState в web запросе	dbo.state
Поставлен на охрану + замкнут	ON,ARMED	ON ARMED
Поставлен на охрану + замкнут + тревога	ON,ALARMED	ON ALARMED
Поставлен на охрану + замкнут + тревога принята	ON,CONFIRMED	ON CONFIRMED
Поставлен на охрану + замкнут + обрыв связи	ON,DETACHED_DISARM	ON DETACHED_DISARM
Снят с охраны + замкнут	ON,DISARMED	ON DISARMED
Снят с охраны + замкнут + тревога	ON,ALARMED	ON ALARMED
Снят с охраны + замкнут + тревога принята	ON,CONFIRMED	ON CONFIRMED
Снят с охраны + замкнут + обрыв связи	ON,DETACHED_DISARM	ON DETACHED_DISARM
Поставлен на охрану + разомкнут	ARMED,OFF	ARMED OFF
Поставлен на охрану + разомкнут + тревога	OFF,ALARMED	OFF ALARMED
Поставлен на охрану + разомкнут + тревога принята	OFF,CONFIRMED	OFF CONFIRMED
Поставлен на охрану + разомкнут + обрыв связи	DETACHED_DISARM,OFF	DETACHED_DISARM OFF
Снят с охраны + разомкнут	DISARMED,OFF	DISARMED OFF
Снят с охраны + разомкнут + тревога	OFF,ALARMED	OFF ALARMED
Снят с охраны + разомкнут + тревога принята	OFF,CONFIRMED	OFF CONFIRMED
Снят с охраны + разомкнут + обрыв связи	DETACHED_DISARM,OFF	DETACHED_DISARM OFF

14.6.4 Список доступных действий с объектом, находящимся в определённом состоянии

Список действий запрашивается не по классу объекта, а берётся из контекста конкретного объекта, т.к. возможны различные права пользователя на объекты одного и того же класса. Работа с полученным списком описана в разделе [Отсылка команд на сервер](#).

14.6.4.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/configuration/{objectClass}:{id}/state/actions/

14.6.4.2 Параметры запроса:

Параметр	Обязательный	Описание
objectClass	Да	Название класса объектов
id	Да	Идентификатор объекта

14.6.4.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/configuration/GRAY:2/state/actions/

14.6.4.4 Пример ответа:

XML

```
<actions>
  <action>
    <description>Disarm</description>
    <hidden>>false</hidden>
    <id>DISARM</id>
  </action>
  <action>
    <description>Arm</description>
    <hidden>>false</hidden>
    <id>ARM</id>
  </action>
  <action>
    <description>Classify alarm</description>
    <hidden>>false</hidden>
    <id>CONFIRM</id>
  </action>
</actions>
```

Если состояние объекта не предусматривает никаких действий, то xml будет таким:

```
<actions/>
```

14.6.4.5 JSON

```
[
  {
    "id": "DISARM",
    "description": "Disarm",
    "hidden": false,
    "react": {
      "sourceType": "MACRO",
      "sourceId": "DISARM",
      "action": "RUN",
      "params": {},
      "name": "React",
      "version": 1
    }
  },
  {
    "id": "ARM",
    "description": "Arm",
    "hidden": false,
    "react": {
      "sourceType": "MACRO",
      "sourceId": "ARM",
      "action": "RUN",
      "params": {},
      "name": "React",
      "version": 1
    }
  },
  {
    "id": "CONFIRM",
    "description": "Classify alarm",
    "hidden": false,
    "react": {
      "sourceType": "MACRO",
      "sourceId": "CONFIRM",
      "action": "RUN",
      "params": {},
      "name": "React",
      "version": 1
    }
  }
]
```

14.6.4.6 Параметры ответа:

Параметр	Описание
description	Текстовое описание реакции
hidden	true – реакция не отображается в интерфейсе (на карте, в макрокомандах и т.д.) false – реакция отображается в интерфейсе

Параметр	Описание
id	Системное название реакции
Блок "react" в ответе JSON	
sourceType	Тип объекта, который может запустить действие.
sourceId	Системное название реакции
action	Действие над объектом sourceType
params	Доступные параметры реакции
name	Тип действия: Event или React
version	Версия

14.6.5 Получение списка всех областей и регионов

14.6.5.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/configuration/zones

14.6.5.2 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/configuration/zones

14.6.5.3 Пример ответа:

JSON:

```
[
  {
    "id": "1",
    "name": "Область 1",
    "regions": [
      {
        "id": "1.1",
        "zoneId": "1",
        "name": "Раздел 1.1",
        "zoneDescription": "Описание области"
      }
    ]
  },
  {
    "id": "2",
    "name": "Область 2",
    "regions": [
      {
        "id": "2.1",
        "zoneId": "2",
```

```

        "name": "Раздел 2.1",
        "zoneDescription": "Описание области"
    }
  ]
}
]

```

XML:

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<zones>
  <zone>
    <id>1</id>
    <name>Область 1</name>
    <regions>
      <id>1.1</id>
      <name>Раздел 1.1</name>
      <zoneDescription>Описание области</zoneDescription>
      <zoneId>1</zoneId>
    </regions>
  </zone>
  <zone>
    <id>2</id>
    <name>Область 2</name>
    <regions>
      <id>2.1</id>
      <name>Раздел 2.1</name>
      <zoneDescription>Описание области</zoneDescription>
      <zoneId>2</zoneId>
    </regions>
  </zone>
</zones>

```

14.6.5.4 Параметры ответа:

Параметр	Описание
id	Идентификатор области/раздела
name	Название области/раздела в ПК <i>Интеллект</i>
zoneDescription	Описание области

14.7 Получение событий

Получение событий от камер, которые изменили своё состояние во время выполнения запроса (~30-60 секунд).

14.7.1 Общий формат запроса:

<http://IP-адрес:порт/web2/secure/feed/>

14.7.2 Пример запроса:

http://127.0.0.1:8085/web2/secure/feed/

14.7.3 Пример ответа:

```
<message>
  <action>update</action>
  <objectId>CAM:1</objectId>
  <state>disconnected</state>
</message>

<message>
  <action>state</action>
  <objectId>CAM:1</objectId>
  <x>10.0</x>
  <y>123.9</y>
</message>

<message>
  <action>state</action>
  <objectId>CAM:1</objectId>
  <state>connected</state>
  <x>300.8</x>
  <y>670</y>
</message>

<message>
  <action>state</action>
  <objectId>CAM:1</objectId>
  <x>100</x>
  <y>100</y>
</message>

<message>
  <action>ping</action>
</message>
```

14.7.4 Параметры ответа:

Параметр	Описание
action	Тип события. Возможные значения: create, delete, update.
objectId	id объекта, от которого приходит событие (обязательно приходит с update, delete, create).
state	id нового состояния объекта (обязательно приходит в create. Если состояние не изменилось, то в событии update состояния не будет).
x, y	Новые координаты, если изменились.
ts	Дата и время изменения состояния объекта.

14.7.5 Получение событий видеоподсистемы блоками

14.7.5.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/events/

14.7.5.2 Параметры запроса:

Параметр	Описание
from	Самая старая дата промежутка поиска сообщений. Пример: 2012-12-27T15%3A19%3A16.000%2B03%3A00
to	Самая последняя дата промежутка поиска сообщений. Пример: 2012-12-27T15%3A19%3A16.000%2B03%3A00
count	Максимальное количество сообщений в ответе в диапазоне [1, 200]. По умолчанию 20. Сервер может вернуть чуть больше, если сообщений в базе данных осталось мало.
objectId	Класс объекта и Id объекта, который указывается через двоеточие. Примеры: CAM:1, GRAY:5, PEOPLE_COUNTER:1 и т.д. События можно получать с нескольких объектов, указав их через запятую. Пример: objectId=CAM:1,PEOPLE_COUNTER:1 – вернуться события по камере 1 и по счётчику 1. Если в параметре не задан Id объекта, то возвращаются события всех объектов заданного класса. Пример: objectId=CAM – вернуться события по всем камерам.
action	Тип события. Если параметр задан, то в ответе будут события только заданного типа. Примеры: <ul style="list-style-type: none"> • REC – начало записи • REC_STOP – остановка записи • ARM – постановка камеры на охрану • DISARM – снятие камеры с охраны • disconnected – потеря связи с камерой

14.7.5.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/events?

from=2021-05-26T14%3A30%3A30.000%2B03%3A00&to=2021-05-26T15%3A40%3A30.000%2B03%3A00&count=5&objectId=MACRO:7

14.7.5.4 Пример ответа:

XML:

```
<events>
  <event>
    <description>Action executed</description>
    <id>{3AED63A0-19BE-EB11-9020-B42E99FDB342}</id>
    <objectId>MACRO:7</objectId>
    <addInfo>event1</addInfo>
    <params0>User 1</params0>
    <params1>1</params1>
    <params2/>
    <params3/>
    <ts>2021-05-26T14:58:05+03:00</ts>
```

```

    <type>Empty</type>
  </event>
  <event>
    <description>Action executed</description>
    <id>{26840B9A-19BE-EB11-9020-B42E99FDB342}</id>
    <objectId>MACRO:7</objectId>
    <addInfo>event2</addInfo>
    params0>User 1</params0>
    <params1>1</params1>
    <params2/>
    <params3/>
    <ts>2021-05-26T14:58:04+03:00</ts>
    <type>Empty</type>
  </event>
</events>

```

JSON:

```

[
  {
    "id": "{3AED63A0-19BE-EB11-9020-B42E99FDB342}",
    "objectId": "MACRO:7",
    "ts": "2021-05-26T14:58:05.000+03:00",
    "description": "Action executed",
    "addInfo": "event1",
    "type": "Empty",
    "params2": "",
    "params3": "",
    "params1": "1",
    "params0": "User 1"
  },
  {
    "id": "{26840B9A-19BE-EB11-9020-B42E99FDB342}",
    "objectId": "MACRO:7",
    "ts": "2021-05-26T14:58:04.000+03:00",
    "description": "Action executed",
    "addInfo": "event2",
    "type": "Empty",
    "params2": "",
    "params3": "",
    "params1": "1",
    "params0": "User 1"
  }
]

```

14.7.5.5 Коды возврата:

- 200 - ОК
- 400 - неверный параметр (формат даты, например)
- 500 - ошибка
- 503 - ошибка соединения с ядром
- 504 - таймаут (ядро не вернуло данные в течение 2000 миллисекунд)

14.8 Отсылка команд на сервер

14.8.1 Общий формат запроса:

PUT http://IP-адрес:порт/web2/secure/configuration/{objectClass}:{id}/state/actions/{CMD}/execute

14.8.2 Параметры запроса:

Параметр	Обязательный	Описание
objectClass	Да	Название класса объектов
id	Да	Идентификатор объекта
CMD	Да	Команда <i>Внимание! Название команды должно быть указано в верхнем регистре.</i>

14.8.3 Пример запроса:

PUT http://127.0.0.1:8085/web2/secure/configuration/GRAY:2/state/actions/DISARM/execute

14.9 Макрокоманды

Макрокоманды (макросы) – это некоторая предопределённая последовательность реакций на определённые события. Макрокоманды создаются на сервере и имеют ID и название. Они похожи на действия с объектами, но не привязаны к объекту.

14.9.1 Получение списка макрокоманд

14.9.1.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/actions/

14.9.1.2 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/actions/

14.9.1.3 Пример ответа:

```
<actions>
  <action>
    <description>Начать запись по всем камерам</description>
    <id>2</id>
  </action>
  <action>
    <description>Снять с охраны все зоны</description>
    <id>1</id>
  </action>
</actions>
```

14.9.1.4 Параметры ответа:

Параметр	Описание
description	Название макрокоманды
id	Идентификатор макрокоманды

14.9.2 Получение параметров макрокоманд

14.9.2.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/actions/{id}/

14.9.2.2 Параметры запроса:

Параметр	Обязательный	Описание
id	Да	Идентификатор макрокоманды

14.9.2.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/actions/2/

14.9.2.4 Пример ответа:

```
<action>
  <description>Начать запись по всем камерам</description>
  <id>2</id>
</action>
```

14.9.2.5 Параметры ответа:

Параметр	Описание
description	Название макрокоманды
id	Идентификатор макрокоманды

14.9.3 Запрос на выполнение макрокоманды на сервере

14.9.3.1 Общий формат запроса:

PUT http://IP-адрес:порт/web2/secure/configuration/{id}/state/actions/RUN/execute

14.9.3.2 Параметры запроса:

Параметр	Обязательный	Описание
id	Да	Тип и идентификатор макрокоманды в формате ТИП:ID (например MACRO:1)

14.9.3.3 Пример запроса:

PUT http://127.0.0.1:8085/web2/secure/configuration/MACRO:1/state/actions/RUN/execute

14.10 Видео

14.10.1 Запрос миниатюр (скриншотов)

14.10.1.1 Общий формат запроса:

14.10.1.1.1 1-й способ

GET http://IP-адрес:порт/web2/secure/video/image.jpg?cam.id={cam.id}&width={width}&height={height}&version={version}&login={login}&password={password}

14.10.1.2 Параметры запроса:

Параметр	Обязательный	Описание
cam.id	Да	Идентификатор камеры
width	Нет	Ширина кадра. Значение может быть в диапазоне [64, 1600]. Сервер автоматически округляет ширину до большего значения, кратного 4 Если нет, то размер возвращаемого изображения берётся из видеопотока
height	Нет	Высота кадра. Значение может быть в диапазоне [30, 1200] Если нет, то размер возвращаемого изображения берётся из видеопотока.
version	Нет	См. Версия продукта
login	Нет	Имя пользователя ПК <i>Интеллект</i> , если установлен
password	Нет	Пароль пользователя ПК <i>Интеллект</i> , если установлен

14.10.1.2.1 2-й способ

GET http://IP-адрес:порт/web2/secure/video/action.do?version={version}&command=frame.video&video_in={video_in}&imageWidth={imageWidth}&imageHeight={imageHeight}&login={login}&password={password}

14.10.1.3 Параметры запроса:

Параметр	Обязательный	Описание
version	Нет	См. Версия продукта
video_in	Да	Идентификатор камеры в формате "ТИП:ИДЕНТИФИКАТОР", например, "CAM:1"
imageWidth	Нет	Ширина кадра. Значение может быть в диапазоне [64, 1600]. Сервер автоматически округляет ширину до большего значения, кратного 4 Если нет, то размер возвращаемого изображения берётся из видеопотока
imageHeight	Нет	Высота кадра. Значение может быть в диапазоне [30, 1200] Если нет, то размер возвращаемого изображения берётся из видеопотока.
login	Нет	Имя пользователя ПК <i>Интеллект</i> , если установлен
password	Нет	Пароль пользователя ПК <i>Интеллект</i> , если установлен

14.10.1.4 Пример запроса:

14.10.1.4.1 1-й способ

GET <http://127.0.0.1:8085/web2/secure/video/image.jpg?cam.id=5&width=85&version=4.7.8.0&login=USER&password=PASS>

14.10.1.4.2 2-й способ

GET http://127.0.0.1:8085/web2/secure/video/action.do?version=4.9.0.0&command=frame.video&video_in=CAM:1&imageWidth=400

14.10.1.5 Пример ответа:

В ответ придет jpeg изображение приблизительно запрошенного размера, либо код об ошибке и body нулевой длины (т.е. придут только заголовки).

Если произошла ошибка, то возвращается http код ошибки:

404 – камера отключена или не используется (disabled);

403 – неверный пароль;

426 – старая версия клиента;

429 – слишком много запросов;

444 – потерян сигнал по камере или камера отключена (коаксиальный провод отключен от платы);

503 – ошибка архива.

14.10.2 Запрос конфигурации

14.10.2.1 Общий формат запроса:

GET <http://IP-адрес:порт/web2/secure/video/config.properties?version={version}&login={login}&password={password}>

14.10.2.2 Параметры запроса:

Параметр	Обязательный	Описание
version	Да	См. Версия продукта
login	Нет	Имя пользователя ПК <i>Интеллект</i> , если установлен
password	Нет	Пароль пользователя ПК <i>Интеллект</i> , если установлен

14.10.2.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/video/config.properties?version=4.7.8.0&login=USER&password=PASS

14.10.2.4 Пример ответа:

Текстовый файл config.properties.

Если пароль установлен, но не указан в запросе:

```
password.enabled=true
login.enabled=true
password.invalid=true#
```

Если пароль правильный или доступ разрешен без пароля:

```
password.enabled=true
login.enabled=true
password.invalid=false
cam.0.id=2
cam.0.name=Face
cam.0.rights=11
cam.1.id=3
cam.1.name=Camera 3
cam.1.rights=11
cam.2.id=5
cam.2.name=Camera 5
cam.2.rights=11
cam.2.telemetry_id=1.1
cam.count=3#
```

14.10.2.5 Параметры ответа:

Параметр	Описание
password.enabled	false - пароль не требуется true - нужен пароль
login.enabled	false - логин не требуется true - нужен логин

Параметр	Описание
password.invalid	false - введён верный пароль true - введён неверный пароль
cam.count	Общее количество камер в присланной конфигурации (id начинается с нуля)
cam.N.id	id камеры
cam.N.name	Название камеры
cam.N.rights	Права (они проверяются на сервере, но чтобы не показывать пользователю лишних опций, доступны и на клиенте). Параметр представляет собой флаги. Если флаг проставлен, то элемент интерфейса следует показывать, если нет, то скрывать. static final int RIGHT_VIEW = 0x1; // доступен просмотр живого видео (этот всегда проставлен в 1) static final int RIGHT_CONTROL = 0x2; // управление (телеметрия, постановка и снятие с охраны) static final int RIGHT_CONFIG = 0x4; // зарезервирован static final int RIGHT_HISTORY = 0x8; // доступ к архиву
cam.N.telemetry_id	id телеметрии (может отсутствовать, если телеметрии нет, тогда необходимо скрывать элементы управления телеметрией)

14.10.3 Запрос видео

14.10.3.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/video/action.do?version={version}&sessionid={sessionid}&video_in={video_in}&normalize={normalize}&imageWidth={imageWidth}&imageHeight={imageHeight}&fps={fps}&login={login}&password={password}

14.10.3.2 Параметры запроса:

Параметр	Обязательный	Описание
version	Да	См. Версия продукта . Если указать в запросе версию 4.10.0.0, в результате будет получен поток в формате MJPEG без xml-вставок, который можно отображать на web-странице в браузерах Chrome и FireFox при помощи тэга IMG. Данная функция реализована как для живого, так и для архивного видео. <i>Примечание. Допускается одновременное получение не более 6 потоков видео.</i>
video_in	Да	Идентификатор камеры в формате "ТИП:ИДЕНТИФИКАТОР", например, "CAM:1"
sessionid	Нет	Идентификатор сессии
imageWidth	Нет	Ширина кадра. Значение может быть в диапазоне [64, 1600]. Сервер автоматически округляет ширину до большего значения, кратного 4
imageHeight	Нет	Высота кадра. Значение может быть в диапазоне [30, 1200]. Если нет, то размер возвращаемого изображения берётся из видеопотока

Параметр	Обязательный	Описание
normalize	Нет	true - растягивает изображение, если кадр приходит в некорректных пропорциях
fps	Нет	Частота кадров видео
login	Нет	Имя пользователя ПК <i>Интеллект</i> , если установлен
password	Нет	Пароль пользователя ПК <i>Интеллект</i> , если установлен

14.10.3.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/video/action.do?
version=4.10.0.0&sessionId=1234567890&video_in=CAM:1&imageWidth=200&fps=1&login=USER&password=PASS

14.10.3.4 Пример ответа:

```
<html>
<head/>
<body>

</body>
</html>
```

14.10.4 Curl запрос видео. Формат основного потока

14.10.4.1 Общий формат запроса:

```
curl -v "http://IP-адрес:порт/web2/secure/video/action.do?fps={fps}&imageHeight={imageHeight}&login={login}&normalize={normalize}&password={password}&sessionId={sessionId}&version={version}&video_in={video_in}" --output ~/output file).bin
```

14.10.4.2 Параметры запроса:

Параметр	Обязательный	Описание
version	Да	См. Версия продукта . Если указать в запросе версию 4.10.0.0, в результате будет получен поток в формате MJPEG без xml-вставок, который можно отображать на web-странице в браузерах Chrome и Firefox при помощи тэга IMG. Данная функция реализована как для живого, так и для архивного видео. <i>Примечание. Допускается одновременное получение не более 6 потоков видео.</i>
video_in	Да	Идентификатор камеры в формате "ТИП:ИДЕНТИФИКАТОР", например, "CAM:1"
sessionId	Нет	Идентификатор сессии

Параметр	Обязательный	Описание
imageWidth	Нет	Ширина кадра. Значение может быть в диапазоне [64, 1600]. Сервер автоматически округляет ширину до большего значения, кратного 4. Если нет, то размер возвращаемого изображения берётся из видеопотока
imageHeight	Нет	Высота кадра. Значение может быть в диапазоне [30, 1200]. Если нет, то размер возвращаемого изображения берётся из видеопотока
fps	Нет	Частота кадров видео
normalize	Нет	true - растягивает изображение, если кадр приходит в некорректных пропорциях
login	Нет	Имя пользователя ПК <i>Интеллект</i> , если установлен
password	Нет	Пароль пользователя ПК <i>Интеллект</i> , если установлен
output file name	Да	Имя файла, в который будет осуществляться вывод запрашиваемого видео

14.10.4.3 Пример запроса:

```
curl -v "http://127.0.0.1:8085/web2/secure/video/action.do?
fps=1&imageHeight=360&login=1&normalize=true&password=1&sessionId=A4D98DDE-A535-49E4-9FB5-
FAD441CBBA43&version=4.10.0.0&video_in=CAM:1" --output ~/output.bin
```

14.10.4.4 Пример ответа:

```
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total      Spent    Left     Speed

  0     0     0     0     0     0     0     0  --:--:--  --:--:--  --:--:--    0* Connected to
172.19.2.6 (172.19.2.6) port 8085 (#0)
* Server auth using Basic with user '1'
> GET /web2/secure/video/action.do?
fps=1&imageHeight=360&login=1&normalize=true&password=1&sessionId=A4D98DDE-A535-49E4-9FB5-
FAD441CBBA43&version=4.10.0.0&video_in=CAM:1 HTTP/1.1
> Host: 172.19.2.6:8085
> Authorization: Basic MTox
> User-Agent: curl/7.74.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Set-Cookie: JSESSIONID=ae532304-e6dc-4a19-8c4f-1e10656140f2; Path=/web2; HttpOnly
< Set-Cookie: rememberMe=deleteMe; Path=/web2; Max-Age=0; Expires=Sun, 24-Jan-2021 08:09:52 GMT
< Server: ITV-Intellect-Webserver/4.11.2.3000
< Date: Mon, 25 Jan 2021 08:09:52 GMT
< Cache-Control: no-store, no-cache, must-revalidate, max-age=0
< Connection: close
< Content-Type: multipart/x-mixed-replace;boundary=videoframe
< Pragma: no-cache
< X-CameraID: 1
```

```
< X-SessionID: A4D98DDE-A535-49E4-9FB5-FAD441CBBA43
< Closing connection 0
<
{ [32768 bytes data]
```

14.10.4.5 Параметры ответа:

В заголовке Content-Type указываются следующие параметры:

Параметр	Описание
multipart/x-mixed-replace	Указывает, что каждая следующая часть контента должна заменять собой предыдущую
boundary	Указывает текстовый разделитель между частями контента

14.10.4.6 Пример ответа:

В файле, в который осуществлялся вывод, для каждой части запрашиваемого контента будут содержаться следующие параметры:

```
--videoframe
Content-Type: image/jpeg
Content-Length: 93081
X-Width: 480
X-Height: 360
X-Timestamp: 0.000
X-Time: 2021-01-25T10:06:42.816+03:00
```

14.10.4.7 Параметры ответа:

Параметр	Описание
Content-Type	Тип контента
Content-Length	Размер части контента
X-Width	Ширина изображения
X-Height	Высота изображения
X-Time	Абсолютное время формирования фрейма
X-Timestamp	Относительное время фрейма в секундах (относительно начала потока)

14.10.4.8 Пример ответа:

В случае завершения потока по вине сервера может прийти завершающий пакет:

```

--videoframe
Content-Type: image/jpeg
Content-Length: 106
<video_in>
  <sessionid>A4D98DDE-A535-49E4-9FB5-FAD441CBBA43</sessionid>
  <video_in>CAM:1</video_in>
  <newstate>closed</newstate>
  <errcode>103</errcode>
</video_in>

```

14.10.4.9 Параметры ответа:

Параметр	Описание
sessionid	id сессии (тот же что и при старте)
video_in	Идентификатор камеры в формате "ТИП:ИДЕНТИФИКАТОР", например, "CAM:1"
errcode	Код ошибки: <ul style="list-style-type: none"> • 100 – отсутствие ошибки. • 101 – слишком много подключенных пользователей. • 102 – неверный пароль (пароль, теоретически, могут поменять в любой момент работы). • 103 – видео недоступно. • 104 – старая версия клиента. Обновите версию.

14.10.5 Управление камерой

14.10.5.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/video/action.do?version={version}&sessionid={sessionid}&cam.id={cam.id}&target=CAM&targetid={targetid}&command={command}&login={login}&password={password}

14.10.5.2 Параметры запроса:

Параметр	Обязательный	Описание
version	Да	См. Версия продукта
cam.id	Да	Идентификатор камеры
sessionid	Нет	Идентификатор сессии
target	Да	CAM – идентификатор класса объекта «Камера»
targetid	Да	Соответствует cam.id
command	Да	Команды управления камерой:

Параметр	Обязательный	Описание
		<ul style="list-style-type: none"> Начало записи: REC Остановка записи: REC_STOP Постановка на охрану: ARM Снятие с охраны: DISARM
manual<номер>	Да	Режим ручной записи. Параметр генерируется автоматически для команд начала и остановки записи камеры. <номер> может принимать значения 0 (режим ручной записи не активирован) или 1 (режим ручной записи активирован)
login	Нет	Имя пользователя ПК <i>Интеллект</i> , если установлен
password	Нет	Пароль пользователя ПК <i>Интеллект</i> , если установлен

14.10.5.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/video/action.do?version=4.10.0.0&sessionId=29101F1&cam.id=1&target=CAM&targetid=1&command=REC¶meters=manual<1>

GET http://127.0.0.1:8085/web2/secure/video/action.do?version=4.10.0.0&sessionId=29101F1&cam.id=1&target=CAM&targetid=1&command=REC_STOP¶meters=manual<1>

GET http://127.0.0.1:8085/web2/secure/video/action.do?version=4.10.0.0&sessionId=29101F1&cam.id=1&target=CAM&targetid=1&command=ARM

GET http://127.0.0.1:8085/web2/secure/video/action.do?version=4.10.0.0&sessionId=29101F1&cam.id=1&target=CAM&targetid=1&command=DISARM

 [Управление телеметрией](#)

14.10.6 Получение авторизованной ссылки на камеру по token ключу

Для получения заранее авторизованной ссылки на камеру (для получения как живого или архивного видео) необходимо:

- Выполнить запрос для получения токена:
- Выполнить запрос по полученному токену.

14.10.6.1 Общий формат запроса для получения токена:

GET http://IP-адрес:порт/web2/secure/video/action.do?version={version}&sessionId={sessionId}&video_in={video_in}&enable_token_auth={enable_token_auth}&valid_token_hours={valid_token_hours}&login={login}&password={password}

14.10.6.2 Параметры запроса:

Параметр	Обязательный	Описание
version	Да	См. Версия продукта
video_in	Да	Идентификатор камеры в формате "ТИП:ИДЕНТИФИКАТОР", например, "CAM:1"

Параметр	Обязательный	Описание
sessionid	Нет	Идентификатор сессии
targetid	Да	Соответствует cam.id
login	Нет	Имя пользователя ПК <i>Интеллект</i> , если установлен
password	Нет	Пароль пользователя ПК <i>Интеллект</i> , если установлен
enable_token_auth	Да	Для включения авторизации по токenu указать значение "1"
valid_token_hours	Нет	Время действия токена в часах. Максимальное значение - неделя. Токен перестает действовать через указанный период времени Значение по-умолчанию 12 часов

14.10.6.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/video/action.do?
version=4.10.0.0&sessionid=FC126734&video_in=CAM:1&enable_token_auth=1&valid_token_hours=1&login=USER&password=PASS

14.10.6.4 Пример ответа:

```
{
  "path" : "action.do?hmac=GAqUa429sjY2E9jCTpuYeaMqReW3Y7HI"
}
```

14.10.6.5 Параметры ответа:

Параметр	Описание
hmac	Токен

14.10.6.6 Общий формат запроса по полученному токenu через Веб-сервер 2.0:

GET http://IP-адрес:порт/web2/secure/video/action.do?hmac={hmac}

14.10.6.7 Параметры запроса:

Параметр	Обязательный	Описание
hmac	Да	Полученный ранее токен

14.10.6.8 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/video/action.do?hmac=GAqUa429sjY2E9jCTpuYeaMqReW3Y7HI

14.10.6.9 Общий формат запроса по полученному токenu через **Web сервер 1**:

Примечание

Данный способ запроса является устаревшим. Рекомендуется выполнять запрос через модуль **Веб-сервер 2.0**.

GET http://IP-адрес:{port}/action.do?hmac={hmac}

14.10.6.10 Параметры запроса:

Параметр	Обязательный	Описание
port	Да	Номер порта, указанный на панели настройки объекта Web сервер для подключения к HTTP-серверу (см. Задание параметров подключения Клиентов к Web-серверу)
hmac	Да	Полученный ранее токен

14.10.6.11 Пример запроса:

GET http://127.0.0.1:80/action.do?hmac=GAqUa429sjY2E9jCTpuYeaMqReW3Y7HI

14.11 Управление телеметрией

14.11.1 Общий формат запроса:

GET http://{login}:{password}@IP-адрес:порт/web2/secure/video/action.do?version={version}&sessionid={session_id}&cam.id={cam_id}&target=PTZ&targetid={PTZ_device_id}&command={command}&login={login}&password={password}&speed={speed}&preset={preset}

Внимание!

Если логин и пароль установлен, то в запросе их необходимо указывать дважды: перед IP-адресом и в виде параметров. В обоих случаях это один и тот же логин и пароль пользователя ПК *Интеллект*.


14.11.2 Параметры запроса:

Параметр	Обязательный	Описание
version	Да	См. Версия продукта
session_id	Да	Идентификатор сессии
cam_id	Да	Идентификатор камеры
PTZ_device_id	Да	Идентификатор устройства управления телеметрией, связанного с камерой (можно получить при запросе конфигурации – см. Получение списка всех объектов сервера)

Параметр	Обязательный	Описание
command	Да	Выполняемая команда. Может принимать следующие значения: <ul style="list-style-type: none"> RIGHT – поворот направо UP – перемещение вверх LEFT – поворот налево DOWN – перемещение вниз ZOOM_IN – увеличение масштаба ZOOM_OUT – уменьшение масштаба GO_PRESET – перейти в определенный пресет POINTMOVE – зуммирование выделенной точки на изображении (x, y) AREAZOOM – зуммирование выделенной области изображения (x,y,w,h)
login	Нет	Имя пользователя ПК <i>Интеллект</i> , если установлен
password	Нет	Пароль пользователя ПК <i>Интеллект</i> , если установлен
speed	Да	Скорость отработки команды (от 0 до 10). При управлении по сети из-за задержек лучше использовать низкие значения
preset	Нет	Номер пресета (число). Обязательный параметр только для command=GO_PRESET. В остальных случаях его значение игнорируется. x – координата x относительно размера изображения. Может принимать значения от 0.0 до 1.0. Обязательный параметр только для command=POINTMOVE или command=AREAZOOM. В остальных случаях его значение игнорируется. y – координата y относительно размера изображения. Может принимать значения от 0.0 до 1.0. Обязательный параметр только для command=POINTMOVE или command=AREAZOOM. В остальных случаях его значение игнорируется. w – ширина области зуммирования относительно размера изображения. Может принимать значения от 0.0 до 1.0. Обязательный параметр только для command=AREAZOOM. В остальных случаях его значение игнорируется. h – высота области зуммирования относительно размера изображения. Может принимать значения от 0.0 до 1.0. Обязательный параметр только для command=AREAZOOM. В остальных случаях его значение игнорируется.

14.11.3 Пример запроса:

```
GET http://user:pass@127.0.0.1:8085/web2/secure/video/action.do?version=4.10.0.0&cam.id=5&target=PTZ&targetid=1.1&command=RIGHT&login=USER&password=PASS&speed=2
```

 Управление камерой

14.12 Работа с архивом

Поток из видеоархива присылается в таком же формате, что и живое видео.

14.12.1 Получение списка записей (1-й способ)

Общий формат запроса:

```
GET http://IP-адрес:порт/web2/secure/video/action.do?version={version}&sessionid={sessionid}&video_in={video_in}&command=arc.intervals&time_from={time_from}&time_to={time_to}&max_count={max_count}&split_threshold={split_threshold}&login={login}&password={password}
```

14.12.1.1 Параметры запроса:

Параметр	Обязательный	Описание
version	Да	См. Версия продукта
sessionid	Нет	Идентификатор сессии
video_in	Да	Идентификатор камеры в формате "ТИП:ИДЕНТИФИКАТОР", например, "CAM:1"
command	Да	Команда для получения списка записей: arc.intervals
time_from	Да	Начало интересующего диапазона времени
time_to	Нет	Конец интересующего диапазона времени
max_count	Нет	Максимальное количество записей в ответе
split_threshold	Нет	Время для объединения нескольких интервалов (в секундах). Интервалы, расстояние между которыми будет меньше заданного, будут объединены в один
login	Нет	Имя пользователя ПК <i>Интеллект</i> , если установлен
password	Нет	Пароль пользователя ПК <i>Интеллект</i> , если установлен
format	Нет	Задаёт формат ответа (см. Общие сведения о HTTP API)

14.12.1.2 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/video/action.do?
 version=4.9.0.0&sessionid=29101F1&video_in=CAM:5&command=arc.intervals&time_from=2013-03-20T00:00:00.000+04:00&time_to=2013-03-22T23:59:59.999+04:00&max_count=100&split_threshold=10399&login=USER&password=PASS

14.12.1.3 Пример ответа:

XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<records count="1" complete="YES" sort="INCREASE">
  <record>
    <from>2011-09-01T00:00:00-05:00</from>
    <to>2011-09-01T00:00:35-05:00</to>
  </record>
  <record>
    <from>2011-09-01T00:00:35-05:00</from>
    <to>2011-09-01T00:01:10-05:00</to>
  </record>
</records>
```

JSON:

```
{ 'count' : 1,
  'complete' : 'YES',
  'sort' : 'INCREASE',
  'cam' : '1',
  'records' : [ {
    'from' : '2019-01-22T12:41:10.144+03:00',
    'to' : '2019-01-23T08:28:47.346+03:00'
  } ]
}
```

14.12.2 Получение списка записей (2-й способ)

14.12.2.1 Общий формат запроса:

GET [http://IP-адрес:порт/web2/secure/archive/{CAM:id}/{DATE}/?\[splitThreshold={splitThreshold}\]&\[days={days}\]](http://IP-адрес:порт/web2/secure/archive/{CAM:id}/{DATE}/?[splitThreshold={splitThreshold}]&[days={days}])

14.12.2.2 Параметры запроса:

Параметр	Обязательный	Описание
CAM:id	Да	Идентификатор камеры в формате "ТИП:ИДЕНТИФИКАТОР", например, "CAM:1"
DATE	Да	Дата начала получения архива. Дата интерпретируется как локальное время для сервера
splitThreshold	Да	Если разница между окончанием предыдущей записи и началом следующей меньше этого числа (в миллисекундах), то записи объединяются в одну. Чтобы никакие записи не объединялись, нужно указать splitThreshold=0. [default: 50]
days	Да	Количество дней от текущего дня, за которые требуется получить архив. [default: 1]

14.12.2.3 Пример запроса:

GET [http://127.0.0.1:8085/web2/secure/archive/CAM:2/2011-12-30/?\[splitThreshold=50\]&\[days=1\]](http://127.0.0.1:8085/web2/secure/archive/CAM:2/2011-12-30/?[splitThreshold=50]&[days=1])

GET <http://127.0.0.1:8085/web2/secure/archive/CAM:1/2013-11-18/?splitThreshold=2000> – получить записи за 18 ноября 2013 года и слепить все записи, промежуток между которыми меньше 2000 миллисекунд.

GET <http://127.0.0.1:8085/web2/secure/archive/CAM:1/2013-11-18/?days=10> – получить записи за 10 дней, начиная с 18 ноября 2013 года.

14.12.2.4 Пример ответа:

XML:

```
<?xml version="1.0" encoding="UTF-16"?>
<days>
  <day>
    <id>2013-11-10T00:00:00-02:00</id>
    <records>
      <from>2013-11-10T18:44:01.579-02:00</from>
```

```

        <to>2013-11-10T18:44:09.717-02:00</to>
    </records>
</day>
<day>
    <id>2013-11-18T00:00:00-02:00</id>
    <records>
        <from>2013-11-18T18:38:30.252-02:00</from>
        <to>2013-11-18T18:38:56.942-02:00</to>
    </records>
    <records>
        <from>2013-11-18T18:39:08.321-02:00</from>
        <to>2013-11-18T18:39:10.080-02:00</to>
    </records>
</day>
</days>

```

JSON:

```

[ {
  "id" : "2013-11-10T00:00:00.000-02:00",
  "records" : [ {
    "from" : "2013-11-10T18:44:01.579-02:00",
    "to" : "2013-11-10T18:44:09.717-02:00"
  } ]
}, {
  "id" : "2013-11-18T00:00:00.000-02:00",
  "records" : [ {
    "from" : "2013-11-18T18:38:30.252-02:00",
    "to" : "2013-11-18T18:38:56.942-02:00"
  }, {
    "from" : "2013-11-18T18:39:08.321-02:00",
    "to" : "2013-11-18T18:39:10.080-02:00"
  } ]
} ]

```

Получение записей за месяц (показывает, в какие дни сентября есть записи):

GET <http://127.0.0.1:8085/web2/secure/archive/CAM:2/2011-09/>

XML:

```

<?xml version="1.0" encoding="UTF-8"?>
<days>
  <day>
    <id>2011-09-02T00:00:00-05:00</id>
  </day>
  <day>
    <id>2011-09-03T00:00:00-05:00</id>
  </day>
  <day>
    <id>2011-09-05T00:00:00-05:00</id>
  </day>
</days>

```

JSON:

```
[ {
```

```

    "id" : "2011-09-01T00:00:00-0500",
    "records" : [ ]
  }, {
    "id" : "2011-09-03T00:00:00-0500",
    "records" : [ ]
  }, {
    "id" : "2011-09-01T00:00:00-0500",
    "records" : [ ]
  } ]

```

Если записей нет, то присылается:

XML:

```

[<days/>
JSON:
[]

```

14.12.3 Получение видео из архива с использованием команды "arc.play"

14.12.3.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/video/action.do?version={version}&sessionid={sessionid}&video_in={video_in}&command=arc.play&time_from={time_from}&time_to={time_to}&login={login}&password={password}&speed_factor={speed_factor}

14.12.3.2 Параметры запроса:

Параметр	Обязательный	Описание
version	Да	См. Версия продукта
sessionid	Нет	Идентификатор сессии
video_in	Да	Идентификатор камеры в формате "ТИП:ИДЕНТИФИКАТОР", например, "CAM:1"
command	Да	Команда для получения видео из архива: arc.play
time_from	Да	Время начала проигрывания архива
time_to	Нет	Время завершения проигрывания архива (если не указано, будет отдан весь архив до последней записи)
imageWidth	Нет	Ширина в пикселях (если не указано или 0, рассчитывается автоматически с сохранением пропорций)
imageHeight	Нет	Высота в пикселях (если не указано или 0, рассчитывается автоматически с сохранением пропорций)
fps	Нет	Максимальная частота кадров в секунду (если не указано или 0, частота кадров не будет ограничиваться)

Параметр	Обязательный	Описание
login	Нет	Имя пользователя ПК <i>Интеллект</i> , если установлен
password	Нет	Пароль пользователя ПК <i>Интеллект</i> , если установлен
speed_factor	Нет	<p>Задаёт скорость воспроизведения. Параметр может принимать любое целое или дробное значение не меньше 0.</p> <p>Примеры значений:</p> <p>0 – воспроизведение с максимально возможной скоростью (зависит от пропускной способности сети и нагрузки на диск)</p> <p>1 – воспроизведение с нормальной скоростью x1 (по умолчанию)</p> <p>0.1 – замедленное воспроизведение со скоростью x1/10</p> <p>2 – ускоренное воспроизведение со скоростью x2</p>
format	Нет	Задаёт формат ответа (см. Общие сведения о HTTP API)

14.12.3.3 Пример запроса:

```
GET http://127.0.0.1:8085/web2/secure/video/action.do?version=4.9.0.0&sessionId=29101F1&video_in=CAM:5&command=arc.play&time_from=2013-03-22T13:04:52.312+04:00&time_to=2013-03-22T13:16:31.873+04:00&login=USER&password=PASS&speed_factor=1
```

14.12.3.4 Пример ответа:

При завершении потока придет завершающий пакет с newstate=closed и errcode=100.

Примечание

Запрос можно выполнять через web или в медиапроигрывателе VLC.

14.12.4 Получение одного кадра из архива с использованием команды "arc.frame"

14.12.4.1 Общий формат запроса (1-й способ):

```
GET http://IP-адрес:порт/web2/secure/video/action.do?version={version}&sessionId={sessionId}&video_in={video_in}&command=arc.frame&time={time}&range={range}&login={login}&password={password}
```

14.12.4.2 Параметры запроса:

Параметр	Обязательный	Описание
version	Да	См. Версия продукта
sessionId	Нет	Идентификатор сессии
video_in	Да	Идентификатор камеры в формате "ТИП:ИДЕНТИФИКАТОР", например, "CAM:1"

Параметр	Обязательный	Описание
command	Да	Команда для получения одного фрейма: arc.frame
time	Да	Время кадра
range	Нет	Время в секундах для задания диапазона поиска ближайшего фрейма относительно time (если не указан, ищется ближайший по всему архиву)
imageWidth	Нет	Ширина в пикселях (если не указано или 0, рассчитывается автоматически с сохранением пропорций)
imageHeight	Нет	Высота в пикселях (если не указано или 0, рассчитывается автоматически с сохранением пропорций)
login	Нет	Имя пользователя ПК <i>Интеллект</i> , если установлен
password	Нет	Пароль пользователя ПК <i>Интеллект</i> , если установлен

14.12.4.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/video/action.do?version=4.9.0.0&sessionId=29101F1&video_in=CAM:5&command=arc.frame&time=2013-03-22T13:04:52.312+04:00&range=0.1&login=USER&password=PASS

14.12.4.4 Общий формат запроса (2-й способ):

GET http://IP-адрес:порт/action.do?version={version}&video_in={video_in}&command=arc.frame&time={time}

14.12.4.5 Параметры запроса:

Параметр	Обязательный	Описание
version	Да	См. Версия продукта
video_in	Да	Идентификатор камеры в формате "ТИП:ИДЕНТИФИКАТОР", например, "CAM:1"
command	Да	Команда для получения одного фрейма: arc.frame
time	Да	Время кадра

14.12.4.6 Пример запроса:

GET http://127.0.0.1:80/action.do?version=4.9.0.0&video_in=CAM:1&command=arc.frame&time=2018-08-12T22:29:06Z

14.12.4.7 Пример ответа:

Request Details		permalink	raw	Headers	
POST	http://webhook.site/4589bc86-e597-41d3-aab8-a93b09e977a8			connection	close
Host	159.69.14.138 whois			x-forwarded-for	159.69.14.138
Date	2019-06-14 12:44:40			user-agent	Apache-HttpClient/4.1.4 (Java/1.8.0_201)
ID	1d1d0274-ddb1-409b-bb75-dc950a12265d			host	webhook.site
Query strings				content-type	application/json; charset=UTF-8
(empty)				content-length	369
Form values				(empty)	
<pre>SubscriptionEvent(action=ALARM_EVENT, uniqueUUID={59321A11-A28E-E911-9D70-1C1B0DE94CF8}, time=Fri Jun 14 15:43:54 MSK 2019, params=Params(additionalDataString= [{"name": "incident", "value": "264400"}, {"name": "picture", "value": "http://172.17.11.11:8095/action.do?version=4.9.0.0&video_in=CAM:8&command=arc.frame&time=2019-06-14T15:43:54Z"}]), cameraCode=8)</pre>					

В обоих случаях в ответ придут http-заголовки и ближайший фрейм из диапазона [time - range, time + range] в формате jpeg. Если фрейма в диапазоне не будет, то тело в ответе будет пустым.

14.12.5 Экспорт архива

14.12.5.1 Создание задания на экспорт архива

По умолчанию экспорт архива осуществляется в формат mp4. Изменить формат можно при помощи ключа реестра ExportContainerFormat (см. [Справочник ключей реестра](#)). Поддерживается экспорт архива, записанного в формате H.264 или MPEG4.

14.12.5.1.1 Общий формат запроса:

POST http://IP-адрес:порт/createarchivetask

```
Content Type: application/json
Content:
{
  "CameraId": "{CameraId}",
  "From": "{From}",
  "To": "{To}"
}
```

14.12.5.1.2 Параметры запроса:

Параметр	Обязательный	Описание
CameraId	Да	Идентификатор камеры
From	Да	Время начала фрагмента архива в формате UTC: ГГГГ-ММ-ДДТЧ:ММ:ССЗ
To	Нет	Время окончания фрагмента архива в формате UTC: ГГГГ-ММ-ДДТЧ:ММ:ССЗ (если не указано, будет отдан весь архив до последней записи)

14.12.5.1.3 Пример запроса:

POST http://127.0.0.1:80/createarchivetask

```
Content Type: application/json
Content:
{
  "CameraId": "1",
  "From": "2016-06-27T15:10:00.00Z",
  "To": "2016-06-27T15:20:00.00Z"
}
```

14.12.5.1.4 Пример ответа:

В папке экспорта (по умолчанию C:\Users\User\Documents) создаётся папка, в названии которой идентификатор задания на экспорт (в примере это 084b56a5-bd49-4327-82db-9bc911f7ff96), внутри – файл в формате mp4.

```
{
  "CameraId" : "1",
  "From" : "2016-06-27T15:10:00.00Z",
  "To" : "2016-06-27T15:20:00.00Z",
  "ArchiveTaskId" : "084b56a5-bd49-4327-82db-9bc911f7ff96",
  "ErrorMessage" : null,
  "State" : "Created"
}
```

14.12.5.1.5 Параметры ответа:

Параметр	Описание
CameraId	Идентификатор камеры
From	Время начала фрагмента архива в формате UTC: ГГГГ-ММ-ДДТЧЧ:ММ:ССZ
To	Время окончания фрагмента архива в формате UTC: ГГГГ-ММ-ДДТЧЧ:ММ:ССZ
ArchiveTaskId	Идентификатор задачи
ErrorMessage	Сообщения об ошибках
State	Результат создания задачи

14.12.5.2 Получение статуса экспорта

14.12.5.2.1 Общий формат запроса:

GET http://IP-адрес:порт/getarchivetaskstatus?archivetaskid={archivetaskid}

14.12.5.2.2 Параметры запроса:

Параметр	Обязательный	Описание
archivetaskid	Да	Идентификатор задачи (см. ответ в Создание задания на экспорт архива)

14.12.5.2.3 Пример запроса:

GET http://127.0.0.1:80/getarchivetaskstatus?archivetaskid=104b38d4-07d7-4d2f-84da-49b3e255d2bf

14.12.5.2.4 Пример ответа:

```
{
  "Percents" : 100,
  "Url" : "http://192.168.15.182:80/download?file=104b38d4-07d7-4d2f-84da-49b3e255d2bf",
  "CameraId" : "1",
  "From" : "2016-06-27T15:10:00.000+03:00",
  "To" : "2016-06-27T15:11:00.000+03:00",
  "ArchiveTaskId" : "104b38d4-07d7-4d2f-84da-49b3e255d2bf",
  "ErrorMessage" : "null",
  "State" : "ReadyForDownload"
}
```

14.12.5.2.5 Параметры ответа:

Параметр	Описание
Percents	Процент выполнения задачи
Url	Ссылка на файл. Требуемый файл можно скачать, используя эту URL ссылку
CameraId	Идентификатор камеры
From	Время начала фрагмента архива в формате ГГГГ-ММ-ДДТЧЧ:ММ:СС
To	Время окончания фрагмента архива в формате ГГГГ-ММ-ДДТЧЧ:ММ:СС
ArchiveTaskId	Идентификатор задачи
ErrorMessage	Сообщения об ошибках
State	Результат выполнения задачи

14.12.5.3 Удаление файла архива

14.12.5.3.1 Общий формат запроса:

DELETE http://IP-адрес:порт/removearchive?archivetaskid={archivetaskid}

14.12.5.3.2 Параметры запроса:

Параметр	Обязательный	Описание
archivetaskid	Да	Идентификатор задачи (см. ответ в Создание задания на экспорт архива)

14.12.5.3.3 Пример запроса:

DELETE http://127.0.0.1:80/removearchive?archivetaskid=084b56a5-bd49-4327-82db-9bc911f7ff96

14.12.5.3.4 Пример ответа:

```
{
  "ArchiveTaskId" : "084b56a5-bd49-4327-82db-9bc911f7ff96",
  "ErrorMessage" : null,
  "Success" : true
}
```

14.12.5.3.5 Параметры ответа:

Параметр	Описание
ArchiveTaskId	Идентификатор задачи
ErrorMessage	Сообщения об ошибках
Success	Результат удаления архива

Из папки экспорта удалится соответствующая папка с файлом.

14.13 Звук

14.13.1 Получение живого звука

14.13.1.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/video/action.do?version={version}&sessionid={sessionid}&command=audio.play&audio_in={audio_in}&format={format}&login={login}&password={password}

14.13.1.2 Параметры запроса:

Параметр	Обязательный	Описание
version	Да	См. Версия продукта
sessionid	Нет	Идентификатор сессии
command	Да	Команда получения живого звука: audio.play
format	Да	Формат аудиоданных

Параметр	Обязательный	Описание
audio_in	Да	Идентификатор микрофона в формате "ТИП:ИДЕНТИФИКАТОР", например, "MIC:1"
login	Нет	Имя пользователя ПК <i>Интеллект</i> , если установлен
password	Нет	Пароль пользователя ПК <i>Интеллект</i> , если установлен

14.13.1.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/video/action.do?
version=4.9.0.0&sessionId=FC126734&command=audio.play&audio_in=MIC:5&format=L16&login=USER&password=PASS

14.13.1.4 Пример ответа:

В ответ будут получены аудиопакеты в следующем виде:

```
HTTP/1.0 200 OK
Connection: close
Server: ITV-Intellect-Webserver/4.9.0.0
Cache-Control: no-store,no-cache,must-revalidate,max-age=0
Pragma: no-cache
Date: Mon, 13 Jan 2013 10:44:27 GMT
Content-Type: multipart/mixed;boundary=audioframe

--audioframe
Content-Type: text/xml
Content-Length: 138

<audio_in>
  <sessionId>FC126734</sessionId>
  <audio_in>MIC:5</audio_in>
  <newstate>started</newstate>
  <errcode>100</errcode>
</audio_in>
--audioframe
Content-Type: audio/L16;rate=8000;channels=1
Content-Length: 1024
X-Time: 2013-03-22T13:16:31.371+04:00

<audio packet PCM16>
--audioframe
Content-Type: audio/L16;rate=8000;channels=1
Content-Length: 1278
X-Time: 2013-03-22T13:16:31.873+04:00
<audio packet PCM16>
```

14.13.2 Остановка получения живого звука

14.13.2.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/video/action.do?version={version}&sessionid={sessionid}&command=audio.stop&audio_in={audio_in}&login={login}&password={password}

14.13.2.2 Параметры запроса:

Параметр	Обязательный	Описание
version	Да	См. Версия продукта
sessionid	Нет	Идентификатор сессии
command	Да	Команда остановки получение живого звука: audio.stop
audio_in	Да	Идентификатор микрофона в формате "ТИП:ИДЕНТИФИКАТОР", например, "MIC:1"
login	Нет	Имя пользователя ПК <i>Интеллект</i> , если установлен
password	Нет	Пароль пользователя ПК <i>Интеллект</i> , если установлен

14.13.2.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/video/action.do?
version=4.9.0.0&sessionid=29101F1&command=audio.stop&audio_in=MIC:5&login=USER&password=PASS

14.13.2.4 Пример ответа:

В этом случае в потоке придёт завершающий xml пакет:

```
--audioframe
Content-Type: text/xml
Content-Length: 106
<audio_in>
  <sessionid>FC126734</sessionid>
  <audio_in>MIC:5</audio_in>
  <newstate>closed</newstate>
  <errcode>100</errcode>
</audio_in>
```

14.13.3 Проигрывание звука из архива

14.13.3.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/video/action.do?version={version}&sessionid={sessionid}&command=arc.play&audio_in={audio_in}&format={format}&time_from={time_from}&time_to={time_to}&login={login}&password={password}

14.13.3.2 Параметры запроса:

Параметр	Обязательный	Описание
version	Да	См. Версия продукта
sessionid	Нет	Идентификатор сессии
command	Да	Команда получения звука из архива: arc.play
format	Да	Формат аудиоданных
audio_in	Да	Идентификатор микрофона в формате "ТИП:ИДЕНТИФИКАТОР", например, "MIC:1"
time_from	Да	Время начала проигрывания архива
time_to	Да	Время завершения проигрывания архива
login	Нет	Имя пользователя ПК <i>Интеллект</i> , если установлен
password	Нет	Пароль пользователя ПК <i>Интеллект</i> , если установлен

14.13.3.3 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/video/action.do?
 version=4.9.0.0&sessionid=29101F1&command=arc.play&audio_in=MIC:5&format=L16&time_from=2013-03-22T13:16:31.873+04:
 00&time_to=2013-03-22T13:04:52.312+04:00&login=USER&password=PASS

14.13.3.4 Пример ответа:

Поток приходит в том же виде, что и при живом звуке (см. [Получение живого звука](#)). При завершении данных приходит завершающий xml пакет (см. [Остановка получения живого звука](#)).

14.13.4 Отправка живого звука

14.13.4.1 Общий формат запроса:

POST http://IP-адрес:порт/web2/secure/video/action.do?version={version}&sessionid={sessionid}
 &command=audio.receive&audio_out={audio_out}&login={login}&password={password}

14.13.4.2 Параметры запроса:

Параметр	Обязательный	Описание
version	Да	См. Версия продукта
sessionid	Да	Идентификатор сессии
command	Да	Команда отправки живого звука: audio.receive

Параметр	Обязательный	Описание
audio_out	Да	Идентификатор спикера в формате "ТИП:ИДЕНТИФИКАТОР", например, "SPEAKER:1"
login	Нет	Имя пользователя ПК <i>Интеллект</i> , если установлен
password	Нет	Пароль пользователя ПК <i>Интеллект</i> , если установлен

14.13.4.3 Пример запроса:

POST http://127.0.0.1:8085/web2/secure/video/action.do?
version=4.9.0.0&sessionId=FC126734&command=audio.receive&audio_out=SPEAKER:3&login=USER&password=PASS

14.13.4.4 Пример ответа:

Отправка звука идёт последовательной передачей пакетов командами:

```
Content-type: audio/L16;rate=8000;channels=1
Connection: keep-alive
```

Далее происходит передача аудиопакета.

14.13.4.5 Параметр ответа:

Параметр	Описание
L16	Формат звука только L16
rate	Любое разумное значение
channels	Канал от 1 до 6

14.14 Пользователи

14.14.1 Получение списка пользователей

14.14.1.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/persons

14.14.1.2 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/persons

14.14.1.3 Пример ответа:

```
{
  "PERSON":
  {
    "external_id":
```

```

{
  "type": "VarChar",
  "value": ""
},
"auto_pass_type":
{
  "type": "Integer",
  "value": ""
},
"rubeg8_zone_id":
{
  "type": "VarChar",
  "value": ""
},
"levels_times":
{
  "type": "LongVarChar",
  "value": ""
},
"expired":
{
  "type": "VarChar",
  "value": ""
},
"card_date":
{
  "type": "VarChar",
  "value": "19.03.2021 15:34:07"
},
"who_level":
{
  "type": "VarChar",
  "value": ""
},
"visit_purpose":
{
  "type": "VarChar",
  "value": ""
},
"level2_id":
{
  "type": "VarChar",
  "value": ""
},
"card":
{
  "type": "VarChar",
  "value": ""
},
"objname":
{
  "type": "VarChar",
  "value":
"\u00CF\u00EE\u00EB\u00FC\u00E7\u00EE\u00E2\u00E0\u00F2\u00E5\u00EB\u00FC 1"
},
"email":
{

```



```
    "type": "VarChar",
    "value": ""
  },
  "area_id":
  {
    "type": "VarChar",
    "value": ""
  },
  "who_card":
  {
    "type": "VarChar",
    "value": ""
  },
  "surname":
  {
    "type": "VarChar",
    "value": ""
  },
  "facility_code":
  {
    "type": "VarChar",
    "value": ""
  },
  "card_loss":
  {
    "type": "Integer",
    "value": ""
  },
  "auto_brand":
  {
    "type": "VarChar",
    "value": ""
  },
  "when_area_id_changed":
  {
    "type": "DBTimeStamp",
    "value": ""
  },
  "post":
  {
    "type": "VarChar",
    "value": ""
  },
  "drivers_licence":
  {
    "type": "VarChar",
    "value": ""
  },
  "temp_levels_times":
  {
    "type": "LongVarChar",
    "value": ""
  },
  "temp_level_id":
  {
    "type": "LongVarChar",
    "value": ""
  }
```

```
},
"temp_card":
{
  "type": "VarChar",
  "value": ""
},
"patronymic":
{
  "type": "VarChar",
  "value": ""
},
"location":
{
  "type": "VarChar",
  "value": ""
},
"teleph_work":
{
  "type": "VarChar",
  "value": ""
},
"department":
{
  "type": "VarChar",
  "value": ""
},
"parent_id":
{
  "type": "VarChar",
  "value": "1"
},
"tabnum":
{
  "type": "VarChar",
  "value": ""
},
"pur":
{
  "type": "Integer",
  "value": ""
},
"finished_at":
{
  "type": "DBTimeStamp",
  "value": ""
},
"all_param":
{
  "type": "VarChar",
  "value": ""
},
"whence":
{
  "type": "VarChar",
  "value": ""
},
"schedule_id":
```

```
{
  "type": "VarChar",
  "value": ""
},
"person":
{
  "type": "VarChar",
  "value": ""
},
"passport":
{
  "type": "VarChar",
  "value": ""
},
"ngp":
{
  "type": "Integer",
  "value": ""
},
"flags":
{
  "type": "Integer",
  "value": ""
},
"auto_number":
{
  "type": "VarChar",
  "value": ""
},
"pin":
{
  "type": "VarChar",
  "value": ""
},
"phone":
{
  "type": "VarChar",
  "value": ""
},
"begin_temp_level":
{
  "type": "VarChar",
  "value": ""
},
"visit_document":
{
  "type": "VarChar",
  "value": ""
},
"visit_birthplace":
{
  "type": "VarChar",
  "value": ""
},
"is_apb":
{
  "type": "SmallInt",
```

```

        "value": "0"
    },
    "end_temp_level":
    {
        "type": "VarChar",
        "value": ""
    },
    "name":
    {
        "type": "VarChar",
        "value":
"\u00CF\u00EE\u00EB\u00FC\u00E7\u00EE\u00E2\u00E0\u00F2\u00E5\u00EB\u00FC 1"
    },
    "started_at":
    {
        "type": "DBTimeStamp",
        "value": ""
    },
    "level_id":
    {
        "type": "LongVarChar",
        "value": ""
    },
    "_marker":
    {
        "type": "LongVarChar",
        "value": ""
    },
    "where_area_id_ap_type":
    {
        "type": "VarChar",
        "value": ""
    },
    "owner_person_id":
    {
        "type": "VarChar",
        "value": ""
    },
    "is_locked":
    {
        "type": "SmallInt",
        "value": "0"
    },
    "is_active_temp_level":
    {
        "type": "SmallInt",
        "value": "0"
    },
    "guid":
    {
        "type": "GUID",
        "value": "{676A1F65-AF88-EB11-9633-1831BF4C3BFA}"
    },
    "card_type":
    {
        "type": "Integer",
        "value": ""
    }

```

```

},
"begin":
{
    "type": "VarChar",
    "value": ""
},
"where_area_id_ap_id":
{
    "type": "VarChar",
    "value": ""
},
"visit_reg":
{
    "type": "VarChar",
    "value": ""
},
"marketing_info":
{
    "type": "LongVarChar",
    "value": ""
},
"comment":
{
    "type": "LongVarChar",
    "value": ""
},
"ad_guid":
{
    "type": "GUID",
    "value": ""
},
" id": "1"
}

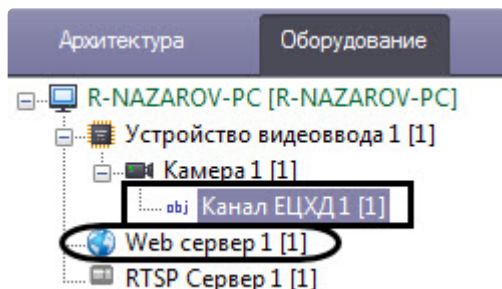
```

14.15 Команды, используемые для интеграции ЕЦХД

ЕЦХД – государственная информационная система «Единый центр хранения и обработки данных».

В данном разделе описаны специальные http-запросы, используемые для интеграции ПК *Интеллект* с ЕЦХД. Для их работы необходимо, чтобы возможность использования таких запросов была включена на этапе настройки программы – см. [Выбор rtsp сервера для обработки запросов ЕЦХД](#).

Запросы отправляются на **Web сервер** и выполняются только для тех камер, на базе которых создан объект **Канал ЕЦХД**.



14.15.1 Получение информации об устройстве

14.15.1.1 Общий формат запроса:

GET http://IP-адрес:порт/getdeviceinfo

14.15.1.2 Пример запроса:

GET http://127.0.0.1:80/getdeviceinfo

14.15.1.3 Пример ответа:

```
{
  "firmware version": "1.2.3 Rev B.",
  "vendor": "Vendor Title Ltd"
  "model": "Device Model",
  "serial_number": "12345ABCDEF",
  "ptz-status": "not supported"
}
```

14.15.1.4 Параметры ответа:

Параметр	Описание
firmware version	Версия прошивки. Также можно добавить произвольный префикс с помощью ключа реестра AdditionalVersionString – см. Справочник ключей реестра
vendor	Производитель
model	Модель
serial_number	Серийный номер
ptz-status	Поддержка PTZ

14.15.2 Получение списка идентификаторов камер

14.15.2.1 Общий формат запроса:

GET http://IP-адрес:порт/getcameras

14.15.2.2 Пример запроса:

GET http://127.0.0.1:80/getcameras

14.15.2.3 Пример ответа:

```
{
  "cameras": [
```

```

{
  {
    "id": 1,
    "channel": 1,
    "status": "working"
  },
  {
    "id": 2,
    "channel ": 2,
    "status": "signaIJust"
  }
}

```

14.15.2.4 Параметры ответа:

Параметр	Описание
id	Идентификатор камеры
channel	Канал камеры
status	Статус камеры

14.15.3 Диапазоны доступных архивных записей

14.15.3.1 Общий формат запроса:

GET http://IP-адрес:порт/getarchiveranges?cameraid={cameraid}&max_count={max_count}

или

GET http://IP-адрес:порт/getavailablearchiveranges?cameraid={cameraid}&max_count={max_count}

14.15.3.2 Параметры запроса:

Параметр	Обязательный	Описание
cameraid	Да	Идентификатор камеры
max_count	Нет	Максимальное количество записей в ответе. По умолчанию выводятся все записи

14.15.3.3 Пример запроса:

GET http://127.0.0.1:80/getarchiveranges?cameraid=1

14.15.3.4 Пример ответа:

Возвращает периоды времени, за которое доступны архивные записи с указанного средства видеонаблюдения. По умолчанию фрагменты в ответе на запрос «склеиваются» (объединяются), если расстояние между ними составляет менее 5 секунд. Изменить данный период времени можно при помощи ключа реестра **SplitArchiveIntervals** (см. [Справочник ключей реестра](#)).

```
{
  "cameraid": 1,
  "ranges": [
    {
      "from": 1412121600, //unixtime
      "to": 1412172000
    },
    {
      "from": 1412186400,
      "to": 1412188200
    }
  ]
}
```

14.15.4 Работа с видеопотоками

14.15.4.1 Запрос GetLiveUrl

Запрос GetLiveUrl возвращает rtsp URL «живого» видеопотока для указанной камеры.

14.15.4.1.1 Общий формат запроса:

GET http://IP-адрес:порт/getliveurl?cameraid={cameraid}

14.15.4.1.2 Параметры запроса:

Параметр	Обязательный	Описание
cameraid	Да	Идентификатор камеры

14.15.4.1.3 Пример запроса:

GET http://127.0.0.1:80/getliveurl?cameraid=1

14.15.4.1.4 Пример ответа:

```
{
  "rtspurl": "rtsp://device-address/somelivemediastream0"
}
```

14.15.4.2 Запрос GetArchiveURL

GetArchiveURL – возвращает rtsp URL архивного видеопотока, отдаваемого регистратором для указанной камеры, начиная с fromdatetime (и, опционально, заканчивая todatetime).

14.15.4.2.1 Общий формат запроса:

GET http://IP-адрес:порт/getarchiveurl?cameraid={cameraid}&fromdatetime={fromdatetime}&todatetime={todatetime}

14.15.4.2.2 Параметры запроса:

Параметр	Обязательный	Описание
cameraid	Да	Идентификатор камеры
fromdatetime	Да	Время начала фрагмента архива в формате ГГГГ-ММ-ДДТЧЧ:ММ:СС
todatetime	Нет	Время окончания фрагмента архива в формате ГГГГ-ММ-ДДТЧЧ:ММ:СС (если не указано, будет отдан весь архив до последней записи)

14.15.4.2.3 Пример запроса:

GET http://127.0.0.1:80/getarchiveurl?cameraid=1&fromdatetime=2014-10-01T00:00:00&todatetime=2014-10-01T01:20:05

14.15.4.2.4 Пример ответа:

```
{
  "rtspurl": "rtsp://deviceaddress/somearchivemediastream?somedatetimetoken"
}
```

14.15.5 Выгрузка архивов

14.15.5.1 Общий формат запроса:

GET http://IP-адрес:порт/downloadarchivefile?cameraid={cameraid}&fromdatetime={fromdatetime}&todatetime={todatetime}

14.15.5.2 Параметры запроса:

Параметр	Обязательный	Описание
cameraid	Да	Идентификатор камеры
fromdatetime	Да	Время начала фрагмента архива в формате ГГГГ-ММ-ДДТЧЧ:ММ:СС
todatetime	Да	Время окончания фрагмента архива в формате ГГГГ-ММ-ДДТЧЧ:ММ:СС

14.15.5.3 Пример запроса:

GET http://127.0.0.1:80/downloadarchivefile?cameraid=1&fromdatetime=2014-10-01T00:00:00&todatetime=2014-10-01T01:20:05

14.15.5.4 Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/octet-stream
```

Также в результате выполнения команды будет получен файл с расширением es (например, Camera[4] (2019-08-13T11_00_00 - 2019-08-13T12_10_00).es). Чтобы воспроизвести этот файл, необходимо конвертировать его с помощью утилиты ffmpeg. Загрузить эту утилиту можно на официальном сайте <https://ffmpeg.org/>

Команда для конвертации файла в кодеке .264:

```
ffmpeg -i "C:\путь до файла в кодеке .264\Camera[5].es" -c:v copy -bsf:v h264_mp4toannexb -c:a copy -f avi output.avi
```

Команда для конвертации файла в кодеке .265:

```
ffmpeg -i "C:\путь до файла в кодеке .265\Camera[5].es" -c:v copy -bsf:v hevc_mp4toannexb -c:a copy -f avi output.avi
```

В результате выполнения команды будет создан файл в формате .avi (output.avi).

14.15.6 Экспорт архивных записей ЕЦХД

Для экспорта архива ЕЦХД используются команды, описанные в разделе [Экспорт архива](#).

14.15.7 Управление функциями средства видеонаблюдения

Внимание!

Для изменения настроек устройства приведенными ниже командами необходимо, чтобы была отключена функция **Использовать настройки устройства** – см. [Панель настройки объекта Устройство видеоввода](#). Для управления параметрами с помощью команд ЕЦХД необходимо, чтобы сама камера поддерживала соответствующие команды (как при подключении по ONVIF, так и при использовании конкретного драйвера).

14.15.7.1 Общий формат запроса:

```
GET https://IP-адрес:порт/?cameraID={cameraID}&ip={ip}&loqin={loqin}&pass={pass}&action={action}&x={x}&y={y}&z={z}&modelName={modelName}
```

14.15.7.2 Параметры запроса

Параметр	Обязательный	Описание
1	Да	Идентификатор средства видеонаблюдения (камеры).
2	Нет	IP-адрес средства видеонаблюдения.
3	Нет	Учетная запись средства видеонаблюдения.
4	Нет	Пароль доступа к средству видеонаблюдения.
5	Да	Имя команды: degreesmove – дискретное движение. Атомарный сдвиг средства видеонаблюдения в указанном направлении. degreesmove2 – относительное движение. Поворот средства видеонаблюдения относительно текущего положения. Область видимости средства видеонаблюдения делится на сетку, где центральная точка имеет координаты (x:0, y:0), левая верхняя (x:-7, y:7), правая нижняя (x:7, y:-7). Поворот средства видеонаблюдения должен быть осуществлен таким образом, чтобы объект по указанным в команде координатам оказался в центре изображения средства видеонаблюдения. Допускается «оптическая» погрешность, возникающая в результате расстояния до объекта видимости.

Параметр	Обязательный	Описание
		<p>Погрешность, возникающую за счет проекции сферы на плоскость, следует компенсировать.</p> <p>В зависимости от камеры для корректной работы данной команды может быть необходимо установить ключи реестра:</p> <ol style="list-style-type: none"> 1. Камера не поддерживает Point&Click, но поддерживает абсолютные координаты. Ключу реестра ReplacePointAndClick должно быть установлено значение 1 (см. Справочник ключей реестра). 2. Камера поддерживает Point&Click. Ключу реестра ReplacePointAndClick должно быть установлено значение 0, а ключу TelemetryCommandMoveTimeout значение задержки между поворотом и масштабированием в миллисекундах (см. Справочник ключей реестра). <p>setposition – установка положения средства видеонаблюдения. Перевод средства видеонаблюдения в указанное положение в градусах относительно «0» позиции.</p> <p>getposition – получение положение средства видеонаблюдения в плоскостях PAN и TILT в градусах, а также текущие значение зума.</p> <p>focus – команда фокусировки средства видеонаблюдения, где параметр z управляет поведением фокуса:</p> <ul style="list-style-type: none"> • 1: Увеличить фокус • -1 : Уменьшить фокус • 0: Авто <p>iris – команда управления диафрагмой средства видеонаблюдения, где параметр z управляет поведением диафрагмы:</p> <ul style="list-style-type: none"> • 1: Открыть диафрагму • -1 : Закрыть диафрагму • 0: Авто <p>switch_day_night – переключение ночного режима, где параметр z управляет режимом работы средства видеонаблюдения:</p> <ul style="list-style-type: none"> • 1: Дневной режим • -1 : Ночной режим <p>backlight – переключение подсветки, где параметр z управляет режимом работы подсветки:</p> <ul style="list-style-type: none"> • 1: Включить • -1: Выключить <p>switch_color – переключение черно-белого режима, где параметр z управляет режимом работы средства видеонаблюдения:</p> <ul style="list-style-type: none"> • 1: Включить • -1: Выключить
6	Нет	<p>В командах degreesmove, setposition – поворот в плоскости PAN [-180 ..0.. 180].</p> <p>В команде degreesmove2 – поворот в плоскости PAN [-7..0..7].</p> <p>В команде getposition не используется.</p> <p>В командах focus, iris, switch_day_night, backlight, switch_color следует задавать параметру значение 0.</p>
7	Нет	<p>В команде degreesmove, setposition – поворот в плоскости TILT [-180 ..0.. 180].</p> <p>В команде degreesmove2 – поворот в плоскости TILT [-7..0..7].</p> <p>В команде getposition не используется.</p>

Параметр	Обязательный	Описание
		В командах focus , iris , switch_day_night , backlight , switch_color следует задавать параметру значение 0.
8	Нет	В командах degreesmove , setposition – увеличение/уменьшение зума [0.. 100]. В команде degreesmove2 – увеличение/уменьшение зума [-1..0..1]. В команде getposition не используется. В командах focus , iris , switch_day_night , backlight , switch_color – задает режим работы средства видеонаблюдения, см. описание соответствующей команды.
9	Нет	Модель средства видеонаблюдения.

14.15.7.3 Пример запроса:

GET http://127.0.0.1:80/execute?cameraID=7&action=getposition

14.15.7.4 Пример ответа:

Ответ приходит только на команду getposition. Пример в формате JSON:

```
{ "y": 56, "x": 105, "z": 0 }
```

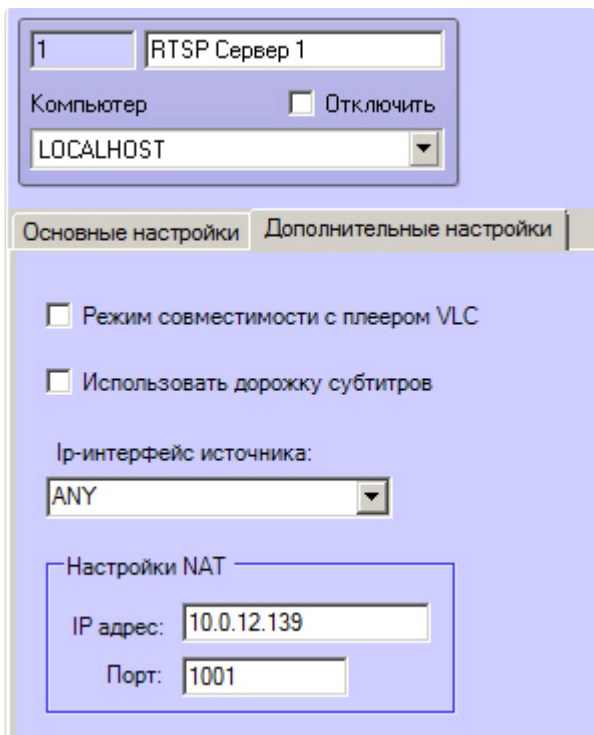
14.15.7.5 Параметры ответа:

Параметр	Описание
x	Координата в плоскости PAN
y	Координата в плоскости TILT
z	Значение масштабирования (зума)

14.15.8 Пример использования команд ЕЦХД при работе с NAT

Включение и конфигурирование использования NAT осуществляется на панели настройки объектов RTSP Сервер и Web сервер (см. [Настройка модуля RTSP Сервер](#) и [Включение обработки запросов ЕЦХД и выбор RTSP сервера](#)).

Например, на панели настройки объекта **RTSP Сервер** указаны следующие параметры NAT:



10.0.12.139 – IP-адрес маршрутизатора.

1001 – порт маршрутизатора.

127.0.0.1 – адрес Сервера в его локальной подсети.

559 – порт RTSP Сервера.

80 – порт Web Сервера.

14.15.8.1 Общий формат запроса:

GET http://IP-адрес:порт/getliveurl?cameraid={cameraid}

14.15.8.2 Параметры запроса:

Параметр	Обязательный	Описание
cameraid	Да	Идентификатор камеры

14.15.8.3 Пример запроса:

GET http://127.0.0.1:80/getliveurl?cameraid=1

14.15.8.4 Пример ответа:

Если флажок **Использовать NAT-адрес** на панели настройки объекта **Web сервер** снят:

```
{
  "rtspurl": "rtsp://127.0.0.1:559/archive?id=1"
}
```

Если флажок **Использовать NAT-адрес** на панели настройки объекта **Web сервер** установлен:

```
{
  "rtspurl": "rtsp://10.0.12.139:1001/archive?id=1"
}
```

14.15.8.5 Параметры ответа:

Параметр	Описание
id	Идентификатор камеры

14.16 Отправка реакций, событий и xml-данных в ПК Интеллект через HTTP API

14.16.1 Отправка реакций в ПК Интеллект через HttpListener

Для того, чтобы использовать функцию отправки реакций в ПК *Интеллект* через HttpListener, на вкладке **Оборудование** диалогового окна **Настройка системы** на базе объекта **Компьютер** должен быть создан объект **Устройство видеоввода**, в поле **Тип** которого выбрано **HttpListener**, а в поле **Порт** указан порт подключения. Также на базе данного объекта **Устройство видеоввода** должно быть создано не более 4 объектов **Луч**.

HttpListener позволяет отправить только реакцию на закрытие/открытие нормально открытого/закрытого луча. После выполнения команды через 2 секунды луч будет переведен в нормальное состояние. В ПК *Интеллект* можно настроить макрокоманды или скрипты с использованием события о сработке луча.

14.16.1.1 Общий формат запроса:

POST http://IP-адрес:порт/device/di/{id}

```
{"state": "{state}"}
```

14.16.1.2 Параметры запроса:

Параметр	Обязательный	Описание
Порт	Да	Порт HttpListener (соответствует номеру порта, указанному в настройках Web-сервера)
id	Да	Идентификатор луча: 0/1/2/3
state	Да	Состояние луча: <ul style="list-style-type: none"> • opened – Открыть • closed – Закрыть

14.16.1.3 Пример запроса:

POST http://127.0.0.1:8085/device/di/0

```
{"state": "closed"}
```

14.16.2 Отправка событий об отсканированном штрихкоде

Отправка событий в ПК *Интеллект* от мобильного приложения о полученной информации с отсканированного штрихкода (barcode).

14.16.2.1 Общий формат запроса:

POST http://IP-адрес:порт/web2/secure/barcode/

```
{ "value": "value", "img":"img" }
```

14.16.2.2 Параметры запроса:

Параметр	Обязательный	Описание
value	Да	Номер штрихкода
img	Нет	Идентификатор отсканированного изображения

14.16.2.3 Пример запроса:

http://1:1@127.0.0.1:8085/web2/secure/barcode

```
{ "value": "321", "img":"6835" }
```

14.16.2.4 Пример ответа:

```
Event : WEB2|1|WEB2_BARCODE|slave_id<Computer>,fraction<243>,owner<Tester>,date<03-12-21>,guid_pk<{96CC2A50-3F54-EC11-B93B-F02F74F7DC0D}>,core_global<1>,barcode<321>,img<6835>,time<16:45:53>
```

14.16.3 Отправка реакций и событий в ПК Интеллект по HTTP-запросу

При получении команд описанного вида в ПК *Интеллект* будут генерироваться обычные события и реакции, которые можно по необходимости использовать в скриптах и макрокомандах (см. [Руководство Администратора](#), раздел [Создание и использование макрокоманд](#), а также [Объект Скрипт. Программирование с использованием языка JScript](#)).

14.16.3.1 Общий формат запроса:

Примечание

Для работы запросов необходимо создать объект Web-сервер – см. [Настройка модуля Web-сервер](#). В запросах необходимо указывать номер порта, указанного в настройках Web-сервера.

GET http://IP-адрес:порт/intellect_core/React?command="{react}"
 GET http://IP-адрес:порт/intellect_core/Event?command="{event}"

или (аналогично)

POST http://IP-адрес:порт/intellect_core/React HTTP/1.1

```
{
  "command" : "{react}"
}
```

POST http://IP-адрес:порт/intellect_core/Event HTTP/1.1

```
{
  "command" : "{event}"
}
```

14.16.3.2 Параметры запроса:

Параметры	Обязательный	Описание
command	Да	React – команда в формате ПК <i>Интеллект</i> Event – событие в формате ПК <i>Интеллект</i>

14.16.3.3 Примеры запроса:

Добавление субтитров на видеоизображение с камеры 2 при помощи HTTP-запроса:

```
GET http://127.0.0.1:80/intellect_core/React?command="CAM|2|ADD_SUBTITLES|command<Some text\n!>"
```

Сгенерировать тревогу по камере 2 при помощи HTTP-запроса:

```
GET http://127.0.0.1:80/intellect_core/Event?command="CAM|2|MD_START"
```

ИЛИ (аналогично)

```
POST http://127.0.0.1:80/intellect_core/Event HTTP/1.1
{
  "command" : "CAM|2|MD_START"
}
```

Вызов макрокоманды 1 при помощи HTTP-запроса:

```
GET http://127.0.0.1:80/intellect_core/React?command="MACRO|1|RUN"
```

Начать запись по Камере 1:


```
POST http://127.0.0.1:80/intellect_core/React HTTP/1.1
{
"command" : "CAM|1|REC"
}
```

14.16.4 Пример отправки команд HTTP API из утилиты curl

Для тестирования работы HTTP API можно отправлять команды HTTP API из утилиты curl. Данная свободно распространяемая утилита доступна для загрузки на официальном сайте <https://curl.haxx.se/>.

Для использования утилиты необходимо запустить командную строку Windows и перейти в папку <Директория установки curl>\curl-7.46.0-win64\bin.

14.16.4.1 Пример запроса:

Команда создания задания на экспорт архива на примере запроса ЕЦХД (см. [Экспорт архива](#)):

```
curl -H "Content-Type: application/json" -X POST -d "{ \"CameraId\": \"1\", \"From\": \"2017-12-26T10:58:00.00Z\", \"To\": \"2017-12-26T11:00:00.00Z\" }" http://127.0.0.1:80/createarchivetask
```

14.16.4.2 Пример ответа:

```
{
  "CameraId" : "1", "From" : "2017-12-26T10:58:00.00Z", "To" : "2017-12-26T11:00:00.00Z",
  "ArchiveTaskId" : "084b56a5-bd49-4327-82db-9bc911f7ff96", "ErrorMessage" : null, "State" :
  "Created"
}
```

Примечание

В запросах необходимо указывать порт, заданный в настройках Web-сервера – см. [Настройка модуля Web-сервер](#).

14.16.5 Отправка xml-файла

HTTP API позволяет получать данные в формате xml для дальнейшей обработки в скриптах. Для отправки файла необходимо выполнить POST запрос.

14.16.5.1 Общий формат запроса:

POST http://IP-адрес:порт/intellect_core/{Any}

```
<tag1>
  <tag2>some_data</tag2>
  <tag3>another_data</tag3>
</tag1>
```

14.16.5.2 Параметры запроса:

Параметр	Обязательный	Описание
Any	Да	<p>Может быть указан любой набор допустимых символов, кроме Event и React, например:</p> <ul style="list-style-type: none"> • http://127.0.0.1:8080/intellect_core • http://127.0.0.1:8080/intellect_core/Any • http://127.0.0.1:8080/intellect_core/Custom

Примечание

В запросах необходимо указывать порт, заданный в настройках Web-сервера – см. [Настройка модуля Web-сервер](#).

14.16.5.3 Пример запроса:

POST http://127.0.0.1:8080/intellect_core/Any

```
<tag1>
  <tag2>some_data</tag2>
  <tag3>another_data</tag3>
</tag1>
```

14.16.5.4 Пример ответа:

После получения данных в формате xml в ПК *Интеллект* будет сгенерировано событие вида:

HTTP|1|CUSTOM_EVENT|url</intellect_core/Any>,owner<SLAVE-ID>,data<PG5..90ZT4=>

14.16.5.5 Параметры ответа:

Параметр	Описание
data	Тело запроса (xml в примере выше) в кодировке base64
url	Часть посланного url

14.17 Настройка интеграции с Техносерв

Для работы интеграции с Техносерв необходимо создать и настроить следующие объекты и файлы:

1. **Web-сервер** – см. [Настройка Сервера для подключения Клиентов с помощью модуля Web-сервер](#)
2. **Веб-сервер 2.0** – см. [Настройка Сервера для подключения Клиентов с помощью модуля Веб-сервер 2.0](#)
3. Хотя бы один объект **Пользователь**, добавленный к **Правам пользователя**. Под учетными данными данного пользователя будет происходить подключение к ПК Интеллект. См. [Администрирование прав и полномочий](#).
4. На панели настройки объекта **Веб-сервер 2.0** необходимо включить передачу событий от камеры и от детекторов – см. [Настройка фильтра событий для модуля Веб-сервер 2.0](#)
5. Все камеры, по которым требуется получать события, добавить в настройки объекта **Web-сервер** – см. [Выбор и настройка видеокamer для Web-сервера](#)

6. Заполнить конфигурационный файл ApiBgConfig.xml и поместить его в папку "<Директория установки ПК Интеллект>\Modules\Jetty\" – см. [Заполнение конфигурационного файла ApiBgConfig.xml](#)

См. также [Примеры команд для работы с интеграцией Техносерв](#).

14.17.1 Заполнение конфигурационного файла ApiBgConfig.xml

Файл ApiBgConfig.xml предназначен для настройки логина и пароля для подключения системы Техносерв к ПК Интеллект, а также для сопоставления событий ПК Интеллект кодам событий Техносерв (см. таблицу ниже).

Файл имеет следующий формат:

```
<ApiBgEventsConfiguration>
  <PingTimeout>120000</PingTimeout>
  <MainHost>127.0.0.1</MainHost>
  <Login></Login>
  <Password></Password>
  <ApiBgEventMatch>
    <ProductObjectName>CAM</ProductObjectName>
    <ProductEventName>DETACHED</ProductEventName>
    <ApiBgEventCode>not_available</ApiBgEventCode>
  </ApiBgEventMatch>
  <ApiBgEventMatch>
    <ProductObjectName>CAM</ProductObjectName>
    <ProductEventName>ATTACH</ProductEventName>
    <ApiBgEventCode>available</ApiBgEventCode>
    <Picture>1</Picture>
  </ApiBgEventMatch>
  <ApiBgEventMatch>
    <ProductObjectName>CAM</ProductObjectName>
    <ProductEventName>ALARM</ProductEventName>
    <ApiBgEventCode>264400</ApiBgEventCode>
    <Shapes>1</Shapes>
    <Picture>1</Picture>
  </ApiBgEventMatch>
  <ApiBgEventMatch>
    <ProductObjectName>CAM</ProductObjectName>
    <ProductEventName>REC</ProductEventName>
    <ApiBgEventCode>264409</ApiBgEventCode>
    <Shapes>1</Shapes>
    <Picture>1</Picture>
  </ApiBgEventMatch>
</ApiBgEventsConfiguration>
```

1. Время в миллисекундах, между отправкой PING-событий
<PingTimeout>120000</PingTimeout>
2. IP-адрес, который необходимо возвращать в ответ на запросы системы Техносерв.
<MainHost>127.0.0.1</MainHost>
3. Логин и пароль для подключения к серверу Техносерв и отправки событий (если есть)
<Login></Login> <Password></Password>
4. Описание соответствия события ПК Интеллект и кода события:
<ApiBgEventMatch>
 - <ProductObjectName>CAM_VMDA_DETECTOR</ProductObjectName>
 - <ProductEventName>ALARM</ProductEventName>
 - <ApiBgEventCode>264400</ApiBgEventCode>
 - <ProductObjectIds>
 - <id>2</id>
 - <id>1</id>
 - </ProductObjectIds>
 - <ParamMatch>
 - <ProductEventParamName>num</ProductEventParamName>
 - <ApiBgEventParamCode>NUMBER_OF_DETECTED_PEOPLE</ApiBgEventParamCode>
 - </ParamMatch>
 - <Shapes>1</Shapes>
 - <Picture>1</Picture>

- a. ProductObjectName, ProductEventName – объект и событие ПК Интеллект, например, CAM|MD_START.
- b. ProductObjectIds – id объектов Интеллекта, если требуется задать конкретные детекторы. Может отсутствовать, тогда будут передаваться события по всем детекторам.
- c. ApiBgEventCode – код события, см. ниже.
- d. Shapes – нужно ли передавать поле Shapes.
- e. Picture – нужно ли передавать ссылку на стоп-кадр.

f. ParamMatch - соответствие имени параметра события Интеллекта параметру события. Может отсутствовать.

Коды событий:

Код	Наименование события
1799	Детектор толпы (Обнаружение скопления людей, в том числе в несанкционированных местах)
264393	Детектор оставленных предметов
264383	Подсчет количества людей
264385	Оценка плотности потока людей на значимых для города объектах
264388	Выявление движения человека против потока
264387	Остановка человека в контролируемой зоне (с заданием времени остановки)
264386	Резкое ускорение движения человека
264389	Хаотичное движение человека (праздношатание)
264391	Индексирование событий в условиях дорожного движения (заторы)
264392	Индексирование событий в условиях дорожного движения (массовое скопление автотранспорта)
264390	Индексирование событий в условиях дорожного движения (плотность потока)
264394	Детектор фактов пересечения запрещенной зоны (проезд, проход)
264395	Детектор исчезнувших предметов
264396	Реагирование на проход людей в заданном направлении (вход)
264397	Реагирование на проход людей в заданном направлении (выход)
264399	Реагирование на проход людей в заданном направлении (коридор и т.п.)
264398	Реагирование на проход людей в заданном направлении (переходы)
264400	Появление человека или автомобиля в зоне наблюдения (улицы, площади, перекрестки, парки)
1795	Детектор расфокусировки
1797	Загрязнение объектива видеокамеры
264403	Изменение фона

Код	Наименование события
264402	Сдвиг камеры
1796	Детектор засветки/заслонения
264401	Открытое пламя
287069	Обнаружена цель
siren	Зафиксирована сирена или автомобильная сигнализация
shout	Зафиксирован крик человека
klaxon	Зафиксирован звук клаксона автомобиля
voice	Зафиксирована речь человека
brake	Зафиксирован звук автомобильного тормоза
shockwave	Зафиксирована ударная волна
shock	Зафиксирован хлопок/выстрел/взрыв
PING	Используется для регулярного события с заданной на стороне ИС периодичностью для контроля состояния канала связи между ИС и АПК БГ. Значение заполняется cameraCode=-1.
not_available	Камера недоступна (отсутствует видеопоток или нет связи с камерой и т.п.)
available	Камера доступна

14.17.2 Примеры команд для работы с интеграцией Техносерв

14.17.2.1 Запрос на получение списка камер

14.17.2.1.1 Общий формат запроса:

GET http://IP-адрес:порт/web2/secure/api/bg

14.17.2.1.2 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/api/bg

Примечание.

Если в поле **Дополнительная информация** на панели настройки камеры указать координаты камеры в формате coords:[longitude]:[latitude]:[angle]:[azimuth]:[radius], то эти данные будут возвращены в ответе на запрос в соответствующих параметрах. См. также [Настройка дополнительной информации о камере](#).

14.17.2.1.3 Пример ответа:

```

0:
  cid: "8"
  name: "Камера 8"
  longitude: "50.0825508"
  latitude: "14.4410435"
  angle: "56"
  azimuth: "223"
  radius: "23"
  webviewurl: "http://172.17.11.11:8095/"
  streamHost: "172.17.11.11"
  streamHttpPort: "8095"
  available: false

```

14.17.2.2 Запрос на получение списка подписок

14.17.2.2.1 Общий формат:

GET http://IP-адрес:порт/web2/secure/api/bg/events/subscriptions

14.17.2.2.2 Пример запроса:

GET http://127.0.0.1:8085/web2/secure/api/bg/events/subscriptions

14.17.2.2.3 Пример ответа:

Если список пустой:

```
{"data": [], "status": "success"}
```

```

data: []
status: "success"

```

при наличии подписок:

```

data:
  0:
    callback: "https://webhook.site/4589bc86-e597-41d3-aab8-a93b09e977a8"
    filter:
      action: "RUN"
      type: "MACRO"
      id: "8"
    id: "63eb2852-9780-4085-bf5e-f3e5be875357"
status: "success"

```

14.17.2.3 Запрос на создание подписки

14.17.2.3.1 Общий формат:

http://IP-адрес:порт/web2/secure/api/bg/events/subscriptions

14.17.2.3.2 Пример запроса:

POST http://127.0.0.1:8085/web2/secure/api/bg/events/subscriptions

```
{"callback":"https://webhook.site/26d15078-c405-4918-8f03-4f2a01b7580f","filter":{"action":"RUN","type":"MACRO"},"id":"5"}
```

14.17.2.3.3 Пример ответа:

Request Details		permalink raw	Headers	
POST	http://webhook.site/4589bc86-e597-41d3-aab8-a93b09e977a8		connection	close
Host	159.69.14.138 whois		x-forwarded-for	159.69.14.138
Date	2019-06-14 12:33:14		user-agent	Apache-HttpClient/4.1.4 (Java/1.8.0_201)
ID	7b6f6e05-0dbd-49d1-b1ff-4f2a734e7421		host	webhook.site
			content-type	application/json; charset=UTF-8
			content-length	369
Query strings			Form values	
(empty)			(empty)	
<pre>SubscriptionEvent(action=ALARM_EVENT, uniqueUUID={4C9E2133-9F8E-E911-9D70-1C1B0DE94CFB}, time=Fri Jun 14 15:23:28 MSK 2019, params=Params(additionalDataString= [{"name":"incident","value":"264400"}, {"name":"picture","value":"http://172.17.11.11:8095/action.do?version=4.9.0.0&video_in=CAM:8&command=arc.frame&time=2019-06-14T15:23:28Z"}]), cameraCode=8)</pre>				

14.17.2.4 Запрос на удаление одной подписки

14.17.2.4.1 Общий формат:

DELETE http://IP-адрес:порт/web2/secure/api/bg/events/subscriptions/{sub_id}

14.17.2.4.2 Параметры запроса:

Параметр	Обязательный	Описание
sub_id	Да	Идентификатор подписки

14.17.2.4.3 Пример запроса:

DELETE http://127.0.0.1:8085/web2/secure/api/bg/events/subscriptions/071e0159-5b2c-4bab-8ed7-42397f1b99b8

14.17.2.5 Запрос на удаление всех подписок

14.17.2.5.1 Общий формат:

DELETE http://IP-адрес:порт/web2/secure/api/bg/events/subscriptions/all

14.17.2.5.2 Пример запроса:

DELETE http://127.0.0.1:8085/web2/secure/api/bg/events/subscriptions/all

14.18 Обращение к API вертикальных решений через HTTP API ПК Интеллект

HTTP API ПК *Интеллект* позволяет обращаться к *API Face-Интеллект* и *API Авто-Интеллект* через Веб-сервер 2.0. Для использования этой функции необходимо следующее:

- REST API *Face-Интеллект*:
 - В фильтр Веб-сервера 2.0 должен быть добавлен объект **Сервер распознавания лиц** (см. [Настройка фильтра событий для модуля Веб-сервер 2.0](#))

- В ответе на запрос списка объектов должен присутствовать объект типа FIRSERVER (см. [Получение списка всех объектов сервера](#))
- REST API Авто-Интеллект:
 - В фильтр Веб-сервера 2.0 должен быть добавлен объект **Канал распознавания номеров** (см. [Настройка фильтра событий для модуля Веб-сервер 2.0](#))
 - В ответе на запрос списка объектов должен присутствовать объект типа ULPR (см. [Получение списка всех объектов сервера](#))

14.18.1 Общий формат запроса

POST http://IP-адрес:порт/web2/secure/{vertical_solution}/{function}

14.18.2 Параметры запроса

Параметр	Обязательный	Описание
vertical_solution	Да	Задает вертикальное решение, к API которого необходимо обратиться. Возможные значения: auto – Авто-Интеллект face – Face-Интеллект Если face, то и в URL, и в теле запроса необходимо передавать id Сервера распознавания лиц в ПК Face-Интеллект: POST http://IP-адрес:порт/web2/secure/face/{server_id}/{function}
function	Да	Функция API вертикального решения. См. описание соответствующего API

Примечание.

Параметры JSON, отправляемого в POST-запросе, см. в документации на соответствующую функцию API.

14.18.3 Примеры запросов

POST http://1:1@127.0.0.1:8085/web2/secure/face/1/firserver/ReadPersons

```
{ "server_id": "1", "objectType": "PERSON", "id": [ ], "page": 1, "pageSize": 2 }
```

GET http://1:1@127.0.0.1:8085/web2/secure/auto/lprserver/GetImage/Frames/66FB34A2-1B38-E811-A92F-001A7DDA710E

14.18.4 Примеры ответов

Примеры ответов см. в документации на соответствующее API вертикального решения.

15 HTTP-сервер ПК Интеллект

15.1 Общие сведения о HTTP Сервере

HTTP Server позволяет передавать события из ПК *Интеллект* в сторонние системы, т.е. оповещать клиенты о событиях.

Для работы с модулем *HTTP Server* необходимо, чтобы веб-браузер поддерживал технологию Server-sent events (SSE).

Примечание.

Браузер Internet Explorer не поддерживает технологию SSE.

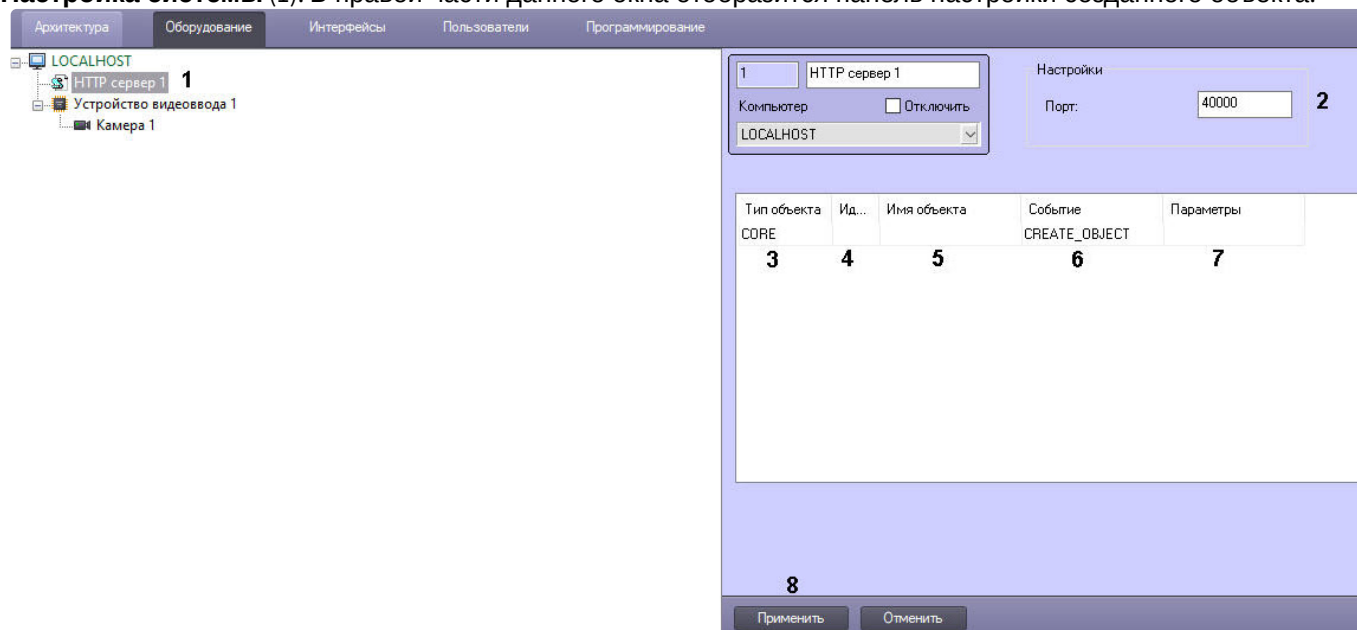
Настройка модуля *HTTP Server* осуществляется на панели настройки соответствующего объекта – см. [Настройка объекта HTTP Сервер](#).

Для работы с модулем *HTTP Server* используются запросы из веб-браузера – см. [Запросы к HTTP Серверу](#).

15.2 Настройка объекта HTTP Сервер

Настройка объекта **HTTP Сервер** осуществляется в следующем порядке:

1. Создать объект **HTTP Сервер** на базе объекта **Компьютер** на вкладке **Оборудование** диалогового окна **Настройка системы** (1). В правой части данного окна отобразится панель настройки созданного объекта.



2. В поле **Порт** указать номер порта в системе, на который клиент должен отправлять запросы к *HTTP Серверу* (2).
3. Настроить фильтр событий, информация о которых должна пересылаться *HTTP Сервером*. Если в фильтр не добавлены никакие события, то *HTTP Сервер* возвращает на запросы клиентов только события ядра (объекта **CORE Ядро**). Выбор дополнительных событий осуществляется следующим образом:
 - a. Из раскрывающегося списка **Тип объекта** выбрать требуемый тип объекта ПК *Интеллект* (3).
 - b. Из раскрывающегося списка **Идентификатор** выбрать идентификатор объекта выбранного типа (4). Если идентификатор не выбран, то *HTTP Сервер* будет отправлять события всех объектов выбранного типа.
 - c. После выбора типа и идентификатора объекта поле **Имя объекта** будет автоматически заполнено названием соответствующего объекта (5).
 - d. В поле **Событие** ввести название события для отправки через *HTTP Сервер* (6). Если событие не указано, то *HTTP Сервер* будет отправлять все события выбранного объекта (объектов выбранного

типа). Название события указывается во внутреннем формате, подробнее см. [Описание событий и реакций объектов системы](#). Например, для того чтобы отфильтровать события постановки камеры на охрану следует выбрать тип объекта **Камера** и ввести в поле **Событие** текст "ARM".

- e. Если требуется отправлять выбранные события, связанные только с пользователями из модуля **Бюро пропусков** (см. [Руководство по настройке и работе с модулем Бюро пропусков](#)), в поле **Параметры** ввести **objtype<PERSON>**. Если этот параметр не задан, события будут отправляться по всем объектам.
 - f. Повторить шаги а-е для всех требуемых объектов и событий.
4. Нажать на кнопку **Применить** (8).

Настройка объекта **HTTP Сервер** завершена.

15.3 Запросы к HTTP Серверу

Для работы с модулем *HTTP Сервер* используются запросы, описанные ниже.

15.3.1 Запрос идентификатора последнего события

GET http://<IP-адрес>:<Порт>/core/GetLastID

Пример запроса:

GET http://localhost:40000/core/GetLastID

Пример ответа:

```
{"lastId":"b686658c-764c-e911-8f42-001a7dda710e"}
```

15.3.2 Запрос событий с учетом фильтра

Фильтр настраивается на панели настройки объекта HTTP Сервер – см. [Настройка объекта HTTP Сервер](#).

GET http://<IP-адрес>:<Порт>/core/Events?keepAliveTime={keepAliveTime}&startingID={startingID}

Параметры запроса:

Параметр	Обязательный	Описание
keepAliveTime	Нет	Интервал времени в секундах, с которым сервер будет отправлять клиенту json вида {"time":"2019-03-19T14:56:26.317"}, содержащий время сервера в стандарте UTC. По умолчанию 5 секунд.
startingID	Нет	Id события, начиная с которого нужно получить события.

Пример запроса:

GET http://localhost:40000/core/Events?keepAliveTime=5&startingID=8FFCDF07-5E4A-E911-95AE-F894C2A95BA4

Пример ответа:

```
{"module":"video.run","protocol_id":"2dc6dfcb-5351-e911-8832-534e57000000","slave_id":"S-UYUTOVA","src_action":"MD_STOP","src_objid":"2","src_objtype":"CAM","time":"2019-03-28T12:20:03.977"}
```

15.3.3 Запрос списка всех созданных считывателей

GET http://<IP-адрес>:<Порт>/core/GetReaders

Пример запроса:

http://localhost:22441/core/GetReaders

с телом

```
{
  "detail": true
}
```

Пример ответа:

```
{
  "objects": [
    {
      "id": "1.1.1",
      "guid": "04EF2B52-90D2-EC11-9515-D8BBC1166DF4",
      "name": "Эмулятор СКУД Считыватель 1.1.1",
      "_address": "1",
      "_marker": "",
      "flags": "",
      "objname": "Эмулятор СКУД Считыватель 1.1.1",
      "objtype": "ACFA_EMULATOR_ACS_READER",
      "parent_id": "1.1",
      "parent_type": "ACFA_EMULATOR_ACS_CONTROLLER",
      "region_in": "",
      "region_out": ""
    },
    {
      "id": "1",
      "guid": "02FA79B0-A5D2-EC11-9515-D8BBC1166DF4",
      "name": "Hikvision Control Reader K1F100 1",
      "_marker": "",
      "card_save_mode": "0",
      "flags": "",
      "objname": "Hikvision Control Reader K1F100 1",
      "objtype": "HIK_CR_K1F100_D8E",
      "parent_id": "localhost",
      "parent_type": "SLAVE"
    },
    {
      "id": "1",
      "guid": "E7065496-A5D2-EC11-9515-D8BBC1166DF4",
      "name": "Контр. считыватель PR-x08 1",
      "_marker": "",
      "card_bits": "16",
      "card_start": "0",
      "fc_bits": "8",
      "fc_start": "16",
      "flags": "",
      "objname": "Контр. считыватель PR-x08 1",
      "objtype": "PARSEC_PR_X08",
      "parent_id": "localhost",
      "parent_type": "SLAVE",
    }
  ]
}
```

```
    "serial_number": ""  
  }  
]  
}
```

16 Интеграция с Axxon Next

Настроенная интеграция ПК *Интеллект* с ПК *Axxon Next* позволяет получать в *Интеллект* видеопотоки и события с камер *Axxon Next*.

Для интеграции используется протокол Onvif.

16.1 Интеграция с Axxon Next по протоколу Onvif

Интеграция ПК *Интеллект* и ПК *Axxon Next* настраивается по порядку в несколько этапов:

1. Обновить ПК *Axxon Next* до версии 4.6.2 совместно с IP Drivers Pack версии 3.74 (это минимально необходимые версии) или новее – см. [Обновление Axxon Next](#), [Обновление модуля Drivers Pack Axxon Next](#), [Обновление ПК Axxon Next в ОС Linux](#).
2. Открыть порт 8081 в ПК *Axxon Next*. Он нужен для работы ONVIF-Сервера.
3. В ПК *Axxon Next* настроить и запустить ONVIF-Сервер – см. [Настройка ONVIF-Сервера](#).
4. Обновить IP Drivers Pack *Интеллекта* до минимально необходимой версии 3.74 или новее. Это можно сделать двумя способами:
 - а. обновить основной продукт до последнего хотфикса, вместе с ним обновится и DriverPack – см. [Обновление программного комплекса Интеллект, Хотфикс-сборки ПК Интеллект и вертикальных решений](#).
 - б. обновить только Drivers Pack — см. [Обновление Drivers Pack отдельно от основного продукта](#).
5. В ПК *Интеллект* настроить Устройство видеоввода определённым образом – см. [Настройка объекта Устройство видеоввода для интеграции с Axxon Next по Onvif](#).

В результате в ПК *Интеллект* будет подключен ONVIF-Сервер *Axxon Next* как Устройство видеоввода с соответствующими ему видеокамерами, которые будут отображаться как дочерние устройства.

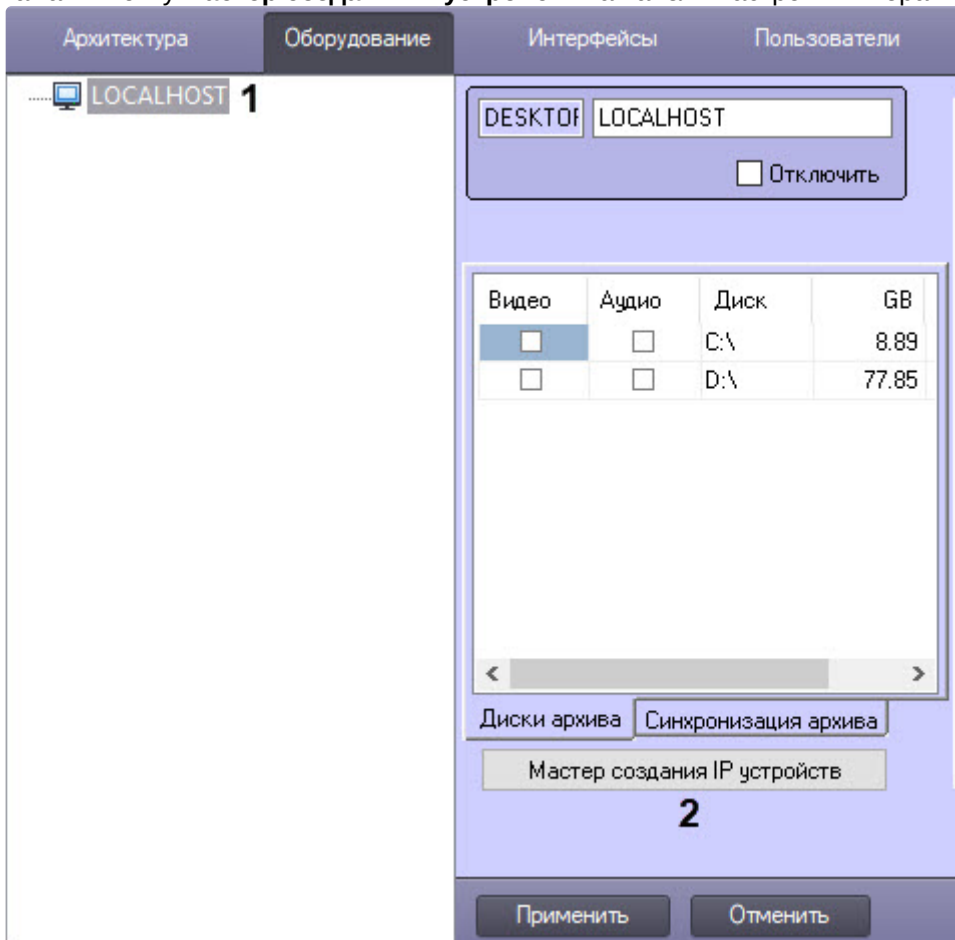
16.1.1 Настройка объекта Устройство видеоввода для интеграции с Axxon Next по Onvif

Есть два варианта настройки объекта **Устройство видеоввода** для получения данных из ПК *Axxon Next*: с помощью **Мастера создания IP-устройств** (предпочтительный вариант) или создание объекта в дереве объектов **Оборудования**. Ниже описаны оба варианта.

16.1.1.1 С помощью Мастера создания IP-устройств

1. Перейти на вкладку **Оборудование** в диалоговом окне **Настройка системы** и в дереве объектов выбрать объект **Компьютер**, соответствующий Серверу, для которого настраивается связь с ПК *Axxon Next* (**1**).

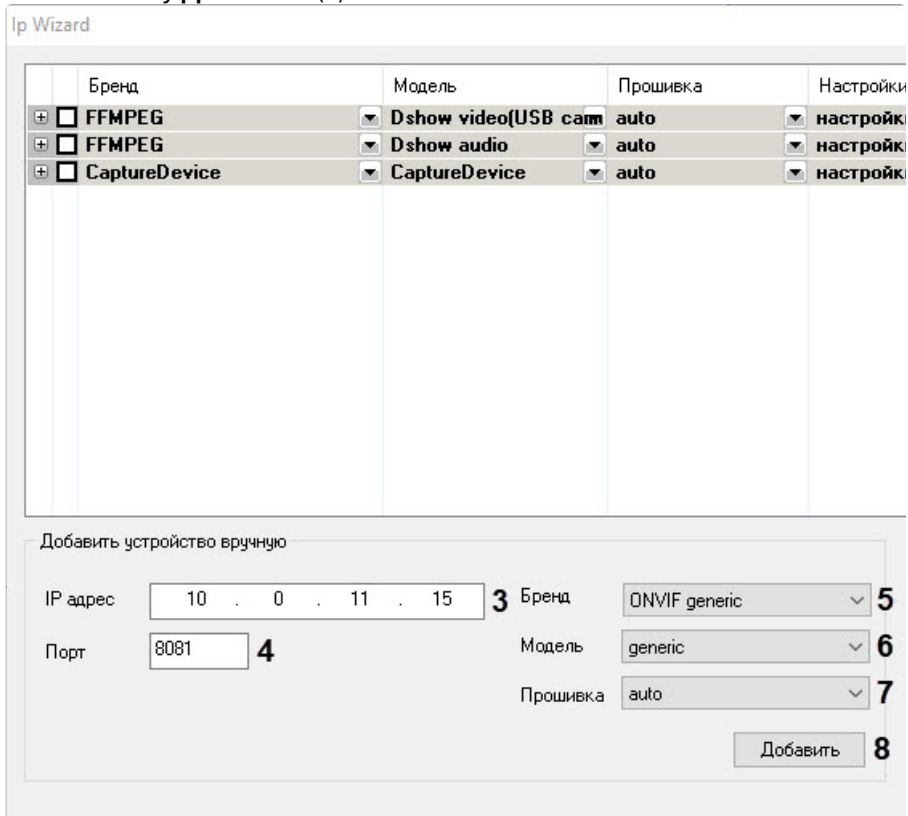
2. Нажать кнопку **Мастер создания IP устройств** на панели настройки выбранного объекта (2).



В результате выполнения операции откроется окно **IP Wizard**.

3. В разделе **Добавить устройство вручную**:
- В поле **IP адрес** ввести IP-адрес ONVIF-Сервера *Axxon Next* (3).
 - В поле **Порт** ввести порт ONVIF-Сервера *Axxon Next* (4).
 - Из списка **Бренд** выбрать **ONVIF generic** (5).
 - Из списка **Модель** выбрать **generic** (6).
 - Из раскрывающегося списка **Прошивка** выбрать **auto** (7).

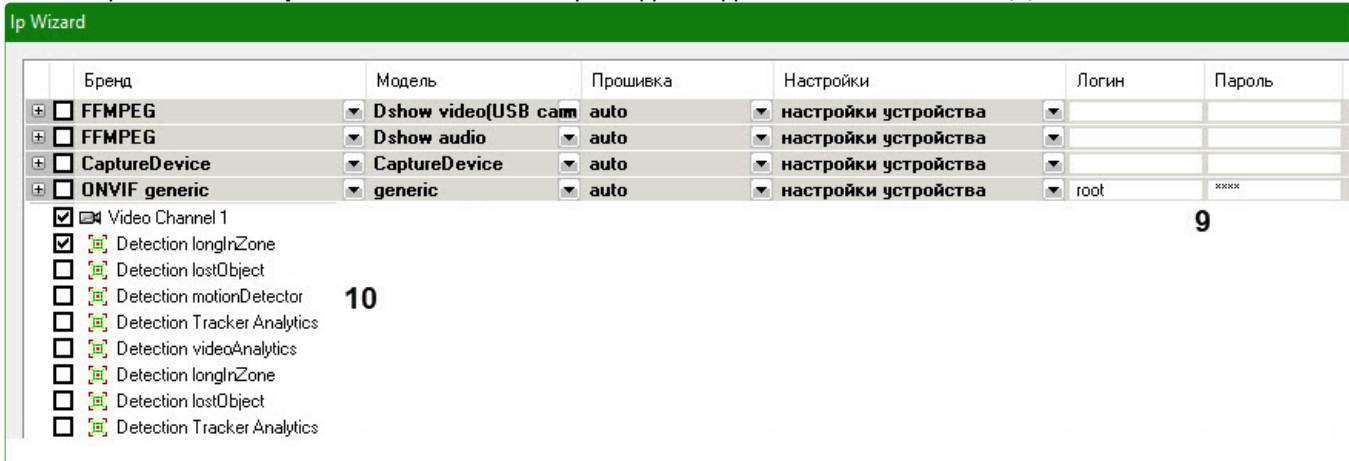
f. Нажать кнопку **Добавить (8)**.



В результате в верхней части окна отобразится IP-устройство с указанными параметрами.

Примечание
 В ПК *Интеллект* версии 4.9 есть отличия в настройке: необходимо выбирать бренд **ONVIF** и модель **4_channel_device**.

4. В столбцах **Логин** и **Пароль** ввести логин и пароль для подключения к *Axxon Next (9)*.



5. Выбрать объекты, которые следует добавить в ПК *Интеллект*. Для этого раскрыть список объектов, нажав кнопку **+**, и установить флажки напротив объектов, которые необходимо создать **(10)**.

Примечание

В ПК *Интеллект* версии 4.9 в качестве детекторов ПК *Axxon Next* будут отображаться **onvif_events**, нужно выбирать их.

Применить

6. Нажать

В дереве объектов вкладки **Оборудование** будут автоматически созданы выбранные объекты.

✔ Полное описание настроек **IP Wizard** – на странице [Мастер создания IP устройств](#).

16.1.1.2 Создание объекта в Оборудовании

1. Создать объект **Устройство видеоввода** на базе объекта **Компьютер** на вкладке **Оборудование** диалогового окна **Настройка системы (1)**.

2. Из раскрывающегося списка **Тип** выбрать значение **ONVIF generic (2)**.

Примечание

В более старых версиях ПК *Интеллект* могут быть типы **ONVIF** или **Onvif(h264)**, они также подходят.

3. В поля **IP-адрес** и **Порт** ввести IP-адрес и порт ONVIF-Сервера *Axxon Next* (3).
4. В полях **Имя** и **Пароль** указать логин и пароль для подключения к *Axxon Next* (4).

5. Нажать

Применить

Созданный таким образом объект **Устройство видеоввода** будет соответствовать Серверу *Axxon Next*. На его базе автоматически отобразятся доступные дочерние объекты, которые есть в *Axxon Next*.

✔ Полное описание настроек **Устройства видеоввода** – на странице [Панель настройки объекта Устройство видеоввода](#).

17 Настройка RabbitMQ

ПК *Интеллект* может выступать в роли приемника и передатчика событий RabbitMQ. Для использования данной функциональности необходимо предварительно загрузить с веб-сайта <https://www.rabbitmq.com/install-windows.html> и установить следующие компоненты:

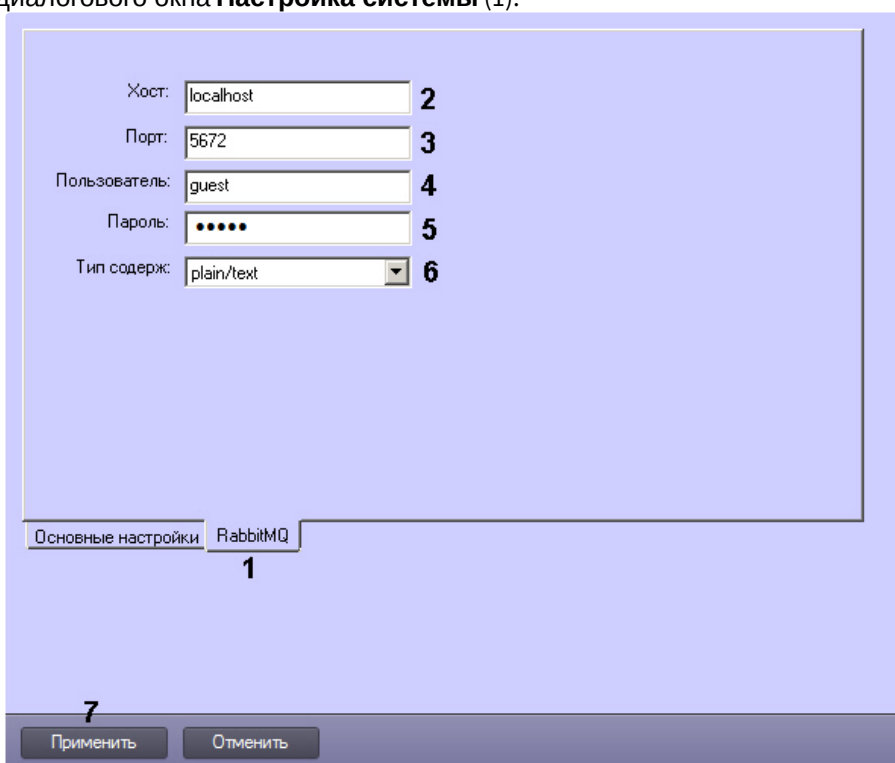
1. Rabbit-сервис
2. Erlang

По умолчанию RabbitMQ имеет встроенную учетную запись **guest/guest**, которая работает только с хостом **localhost**. Если требуется использовать RabbitMQ удаленно, необходимо завести новую учетную запись.

В примерах ниже упоминаются утилиты `amqp_sendstring.exe` и `amqp_listen.exe`. Данные утилиты для обработки и отправки сообщений ПК *Интеллект* программисту необходимо реализовать самостоятельно.

Настройка подключения к RabbitMQ осуществляется следующим образом:

1. Перейти на вкладку **RabbitMQ** на панели настройки **Объект охраны** на вкладке **Программирование** диалогового окна **Настройка системы** (1).



2. В поле **Хост** ввести имя или адрес хоста RabbitMQ (2).
3. В поле **Порт** ввести порт для подключения к хосту RabbitMQ (3). Если порт не задан (по умолчанию) или равен 0, то клиент RabbitMQ не запускается.
4. В поле **Пользователь** ввести имя пользователя хоста RabbitMQ (4).
5. В поле **Пароль** ввести пароль пользователя хоста RabbitMQ (5).
6. Из раскрывающегося списка **Тип содерж.** выбрать тип данных, отправляемых и принимаемых ПК *Интеллект* (6):
 - а. **application/json** – отправляется и принимается сообщение в формате JSON. Если ПК *Интеллект* не может разобрать сообщение в формате JSON, то предпринимается попытка распознать сообщение в формате **plain/text**.
 - б. **plain/text** – отправляется и принимается событие ПК *Интеллект* в формате "ТИП|ИДЕНТИФИКАТОР|СОБЫТИЕ"
7. Нажать на кнопку **Применить** (7).

17.1 Поставщик ПК Интеллект и сторонний приемник

Поставщик ПК *Интеллект* работает в режиме «`amq.topic`», а приемник – в режиме «`amq.direct`».

amq.topic позволяет ранжировать подписки. Каждое событие, отправляемое из ПК *Интеллект*, подписывается специальным заголовком, который строится по следующей схеме:

```
routingkey = "intellect.event." + msg.GetSourceType() + "." + msg.GetSourceId() + "." + msg.GetAction();
```

Таким образом, приемник может гибко подписаться на необходимые события.

Пример.

Подписка на все события:

```
amqp_listen.exe localhost 5672 amq.topic intellect.event.#
```

Подписка только на события с Action=="RUN"

```
amqp_listen.exe localhost 5672 amq.topic intellect.event.*.*.RUN
```

Внимание!

События поступают в очередь RabbitMQ от каждого ядра индивидуально. Например, если в системе два ядра, и событие произошло на ядре, у которого неправильные настройки или нет подключения к RabbitMQ, событие не попадет в очередь, несмотря на то, что второе ядро его получит. Каждое ядро отправляет только свои сообщения.

17.2 Приемник ядра ПК Интеллект

Приемник реализован по схеме amq.direct. Это означает, что он подписан на все события с ключом "bindingkey". bindingkey реализован по схеме "intellect." + ComputerName (чувствителен к регистру).

При отправке bindingkey должен быть указан в точности как у приемника, иначе сообщение не дойдет.

Таким образом, чтобы ядро получило событие, необходимо отправить следующее сообщение:

текстовое

```
amqp_sendstring.exe localhost 5672 amq.direct intellect.ASUS "CAM|1|HELLO"
```

или JSON

```
amqp_sendstring.exe localhost 5672 amq.direct intellect.ASUS {"Type\":"MACRO\","Id\":"1\","Action\":"RUN\","Params\":{"test1\":"+++","\test2\":"000\}}
```

18 Заключение

Более подробная информация о программном комплексе *Интеллект* содержится в следующих документах:

1. [Руководство администратора](#);
2. [Руководство оператора](#);
3. [Руководство по установке и настройке компонентов охранной системы](#);
4. [Руководство по программированию](#);
5. [Руководство по программированию \(JScript\)](#).

Если в процессе работы с данным программным продуктом у вас возникли трудности или проблемы, вы можете связаться с нами. Однако рекомендуем предварительно сформулировать ответы на следующие вопросы:

1. В чем именно заключается проблема?
2. Когда и после чего появилась данная проблема?
3. В каких именно условиях проявляется проблема?

Помните, что чем более полную и подробную информацию вы нам предоставите, тем быстрее наши специалисты смогут устранить вашу проблему.

Мы всегда работаем над улучшением качества своей продукции, поэтому будем рады любым вашим предложениям и замечаниям, касающимся работы нашего программного обеспечения, а также документации к нему.

Пожелания и замечания по данному Руководству следует направлять в Отдел технического документирования компании Ай-Ти-Ви групп (documentation@itv.ru).

19 ПРИЛОЖЕНИЕ 1. Описание структуры ddi-файла

Описание полей таблицы, расположенной на вкладке **Имена** (раздел **<Objects>**), приведено в таблице ниже.

Поле	Описание
Имя (<ObjectName>)	Идентификационное имя объекта
Название (<VisibleName>)	Отображаемое имя
Имя группы (<GroupName>)	Имя группы объектов. Используется для объединения объектов в группу в дереве настроек ПК <i>Интеллект</i>

Описание полей таблицы, расположенной на вкладке **События** (раздел **<Events>**), приведено в таблице ниже.

Поле	Описание
Название (<EventName>)	Идентификационное имя события.
Описание (<EventDescription>)	Описание сообщения, выводимое в протоколе событий.
Обработка сообщений (<EventType>)	Определение цвета фона сообщения в протоколе событий: обычное – без цвета; тревожное – красное окно; информационное – синее окно.
Звуковая поддержка (<IsSoundEnabled>)	Воспроизведение звукового файла при срабатывании сообщения.
Не слать в сеть (<IsNetworkDisabled>)	Не посылать сообщение по сети.
Не протоколировать (<IsProtocolDisabled>)	Не отображать сообщение в протоколе событий и не записывать событие в базу данных.
Журнал Windows (<IsWindowsLogEnabled>)	Сохранять сообщения в журнале событий Windows. <i>Примечание. Запись в журнал Windows невозможна, если событие не протоколируется</i>

Описание полей таблицы, расположенной на вкладке **Реакции** (раздел **<Reacts>**), приведено в таблице ниже.

Поле	Описание
Название (<ReactName>)	Имя реакции
Описание (<ReactDescription>)	Описание реакции, выводимое в контекстном меню при щелчке правой кнопкой мыши по значку объекта на <i>Карте</i>
Флаги (<IsReactArm>)	Флаг выполнения реакции: либо для одного объекта, либо для группы объектов, входящих в один раздел

Описание полей таблицы, расположенной на вкладке **Значки** (раздел **<Icons>**), приведено в таблице ниже.

Поле	Описание
Имя файла (<FileName>)	Часть имени bmp-файла, которая является идентификатором изображения. Идентификатор изображения позволяет использовать несколько bmp-файлов для представления на <i>Карте</i> объектов одного типа (см. раздел Использование утилиты ddi.exe для работы с DDI-файлами)

Название (<IconName>)	Описание bmp-файла объекта
-----------------------	----------------------------

Описание полей таблицы, расположенной на вкладке **Состояния** (раздел **<States>**), приведено в таблице ниже.

Поле	Описание
Название (<StateName>)	Имя состояния
Изображение (<ImgName>)	Часть имени, которая является идентификатором состояния, соответствующего bmp-файла (см. раздел Использование утилиты ddi.exe для работы с DDI-файлами). <i>Примечание. Объекты на Карте могут быть отображены с помощью линий, т.е. без использования bmp-файлов. В этом случае, если изменяется состояние объекта, меняется цвет линии. Цвет (RGB) состоянию задается следующим образом: <Состояние>\$R:G:B</i>
Описание (<StateDescription>)	Описание состояния
Мерцание (<IsStateFlashing>)	Отображение на Карте: обычное – отсутствие мерцания, тревожное – мерцание значка на Карте

Описание полей таблицы, расположенной на вкладке **Правила перехода** (раздел **<Rules>**), приведено в таблице ниже.

Поле	Описание
Событие (<EventName>)	Событие, по которому выполняется переход
Переход из состояния (<FromStateName>)	Исходное состояние, из которого должен быть осуществлен переход
Переход в состояние (<ToStateName>)	Итоговое состояние, в которое должен быть осуществлен переход

20 ПРИЛОЖЕНИЕ 2. Объявление классов NissObjectDLLExt и CoreInterface

На странице:

- [CoreInterface](#)
- [NissObjectDLLExt](#)

20.1 CoreInterface

```

class CoreInterface
{
public:

    virtual BOOL DoReact (React&) = 0;

    virtual BOOL NotifyEvent(Event&) = 0;

    virtual void SetupACDevice(LPCTSTR objtype, LPCTSTR objid, LPCTSTR
objtype_reader) = 0;

    virtual  BOOL    IsObjectExist(LPCTSTR objtype, LPCTSTR id) = 0;
    virtual  BOOL    IsObjectDisabled(LPCTSTR objtype, LPCTSTR id) = 0;
    virtual Msg FindPersonInfoByCard(LPCTSTR facility_code, LPCTSTR card) = 0;
    virtual Msg FindPersonInfoByExtID(LPCTSTR external_id) = 0;
    virtual  CString GetObjectName (LPCTSTR objtype, LPCTSTR id) = 0;
    virtual  CString GetObjectState(LPCTSTR objtype, LPCTSTR id) = 0;
    virtual void      SetObjectState(LPCTSTR objtype, LPCTSTR id, LPCTSTR state)
= 0;
    virtual  BOOL    IsObjectState(LPCTSTR objtype, LPCTSTR id, CString state) =
0;

    virtual  CString GetObjectParam (LPCTSTR objtype, LPCTSTR id, LPCTSTR
param) = 0;
    virtual  int    GetObjectParamInt (LPCTSTR objtype, LPCTSTR id, LPCTSTR param) =
0;

```

```

        virtual CMapStringToStringArray* GetObjectParamList(LPCTSTR objtype, LPCTSTR
id, LPCTSTR param) = 0;

        virtual CStringArray* GetObjectParamList(LPCTSTR objtype, LPCTSTR id,
LPCTSTR param, LPCTSTR name) = 0;

        virtual void GetObjectParams (LPCTSTR objtype, LPCTSTR id, Msg& msg)
= 0;

        virtual void SetObjectParamInt (LPCTSTR objtype, LPCTSTR id, LPCTSTR
param, int val) = 0;

        virtual CString GetObjectIdByParam(LPCTSTR type, LPCTSTR param, LPCTSTR val)
= 0;

        virtual CString GetObjectIdByName(LPCTSTR type, LPCTSTR name) = 0;

        virtual CString GetObjectParentId(LPCTSTR objtype, LPCTSTR id, LPCTSTR
parent) = 0;

        virtual int GetObjectIds(LPCTSTR objtype, CStringArray& list, LPCTSTR main_id
= NULL) = 0;

        virtual int GetObjectChildIds(LPCTSTR objtype, LPCTSTR objid, LPCTSTR
childtype, CStringArray& list) = 0;
};

```

20.2 NissObjectDLLExt

```

class NissObjectDLLExt
{
protected:
    CoreInterface* m_pCore;

public:
    NissObjectDLLExt(CoreInterface* core) { m_pCore = core; }

    virtual CString GetObjectType() = 0;
    virtual CString GetParentType() = 0;
    virtual int GetPos() { return -1; }
    virtual CString GetPort() { return CString(); }
    virtual CString GetProcessName() { return CString(); }
};

```

```

virtual CString GetDeviceType() { return CString(); }

virtual BOOL HasChild() { return FALSE; }

virtual UINT HasSetupPanel() { return FALSE; }

virtual void OnPanelInit(CWnd*) {}

virtual void OnPanelLoad(CWnd*,Msg&) {}

virtual void OnPanelSave(CWnd*,Msg&) {}

virtual void OnPanelExit(CWnd*) {}

virtual void OnPanelButtonPressed(CWnd*,UINT) {}

virtual BOOL IsRegionObject() { return FALSE; }

virtual BOOL IsProcessObject() { return FALSE; }

virtual BOOL IsIncludeParentId() { return 0; }

virtual BOOL IsWantAllEvents() { return 0; }

virtual CString DescribeSubscribeObjectsList() { return CString(); }

virtual CString GetIncludeIdParentType(){ return CString(); }

virtual CString DescribeParamLists(){ return CString(); }

virtual void OnCreate(Msg&) {}

virtual void OnChange(Msg&,Msg&) {}

virtual void OnDelete(Msg&) {}

virtual void OnInit(Msg&) {}

virtual void OnEnable(Msg&) {}

virtual void OnDisable(Msg&) {}

virtual BOOL OnEvent(Event&) { return FALSE; }

virtual BOOL OnReact(React&) { return FALSE; }

};

```