

ДЕЛО



Руководство программиста



ГРУППА КОМПАНИЙ «ЭОС»

Система автоматизации делопроизводства и документооборота «ДЕЛО»

Версия 24.2.0

РУКОВОДСТВО ПРИКЛАДНОГО ПРОГРАММИСТА

Москва
2024

В данном документе приведено описание API - интерфейса (API) системы «ДЕЛО». API предназначен для разработки прикладных программ, формирования произвольных отчетов, справок, получения статистических данных, а также для экспорта данных из системы «ДЕЛО» в другие системы.

В документе содержится описание назначения API системы «ДЕЛО», описание методов, функций, переменных. Приведены примеры прикладных программ, демонстрирующих использование API.

Документ предназначен для прикладных программистов.

Данные, используемые в примерах, рисунках и таблицах настоящего руководства, являются условными.

Книга не является полной документацией к программному обеспечению. Для использования программы необходимо наличие других книг, включаемых в комплект.

Система «ДЕЛО» постоянно совершенствуется и, в связи с этим, возможны некоторые несоответствия, касающиеся описания пользовательского интерфейса.

Документация разработана специалистами ГРУППЫ КОМПАНИЙ «ЭОС».

«ДЕЛО» является торговой маркой ГРУППЫ КОМПАНИЙ «ЭОС».

Право на тиражирование программных продуктов и документации принадлежит ГРУППЕ КОМПАНИЙ «ЭОС».

Все упомянутые в данном издании товарные знаки и зарегистрированные товарные знаки принадлежат своим законным владельцам.

© ГРУППА КОМПАНИЙ «ЭОС». Все права защищены

107113, Москва, ул. Шумкина, д. 20 стр. 1.

Тел/факс: (495) 221-24-31

e-mail: support@eos.ru

<http://www.eos.ru>

СОДЕРЖАНИЕ

1	Архитектура системы «ДЕЛО».....	19
1.1	Модель данных	19
1.1.1	Справочник Группы документов	20
1.1.2	Справочник Подразделения, картотеки и кабинеты.....	21
1.1.3	Справочник Организации	21
1.1.4	РК документа	21
1.1.5	Поручения	22
1.2	Функциональная схема	23
1.3	Технологические стеки	23
1.3.1	ДЕЛО API 2009	24
1.3.2	ДЕЛО API 2023	24
2	OData (для ДЕЛО API 2009).....	26
2.1	Описание протокола.....	26
2.1.1	Идентификатор объекта.....	26
2.1.2	Временный идентификатор объекта.....	26
2.1.3	Представление результата	26
2.1.4	Опции сериализации (\$expand)	27
2.1.5	Указание картотеки и кабинета	27
2.1.6	Параметр criteries.....	27
2.1.7	Пакетные команды	28
2.1.8	Вызов сервисных методов	28
2.2	Функции API.....	29
2.2.1	Загрузка объектов по списку идентификаторов	29
2.2.2	Поиск объектов	29
2.2.3	Модификация объектов	31
2.2.4	Создание объектов	31
2.2.5	Удаление объектов	32
2.2.6	Получение содержимого файла	32
2.2.7	Создание шаблона для новой РК	32
2.2.8	Создание шаблона для нового РКПД	33
2.3	Примеры запросов.....	33
2.3.1	Поиск по номеру документа.....	34
2.3.2	Получение объекта по идентификатору, включая вложенные списки	34
2.3.3	Изменение полей объекта.....	34
2.3.4	Поиск документов по текстовым данным.....	35

2.3.5	Добавление записей в справочник.....	35
2.3.6	Получение объектов по списку идентификаторов.....	36
2.3.7	Получение списков объекта	36
2.4	Объекты системы	37
3	С# ORM EOS (для ДЕЛО API 2009).....	38
3.1	Модель данных, используемая в API фоновых задач.....	38
3.2	Соединение с базой данных	40
3.3	Поиск объектов.....	40
3.4	Поиск ссылки на сущность системы	42
3.5	Загрузка объекта по идентификатору.....	42
3.6	Модификация данных объекта.....	42
3.7	Выполнение хранимых процедур БД	44
4	GraphQL (для ДЕЛО API 2023).....	45
4.1	Описание протокола.....	45
4.1.1	Пример выполнения запроса без параметров.....	45
4.1.2	Пример выполнения запроса с параметрами.....	45
4.1.3	Утилита с графическим интерфейсом для изучения GraphQL (схемы и отладки запросов)	46
4.1.4	Правилами формирования имен сущностей, свойств, методов, запросов.	46
4.2	Запросы GraphQL на получение данных (query).....	47
4.2.1	Пример запроса на получение одной записи:.....	47
4.2.2	Запросы на страничный и курсорный пейджинг.....	47
4.2.3	Запросы с вложенными объектами.....	47
4.2.4	Представление результата	48
4.3	Запросы GraphQL на изменение данных (Mutation)	49
4.3.1	Добавление данных.....	49
4.3.2	Изменение данных	49
4.3.3	Удаление данных.....	50
4.3.4	Вложенные объекты.....	50
4.3.5	Представление результата	51
4.3.6	Кастомные аргументы.....	51
4.3.7	Пакетные команды	52
4.4	Работа с файлами: скачивание файлов.....	52
4.5	Примеры запросов.....	53
4.5.1	Создание Contact и Department	53
4.5.2	Запрос протокола через курсорный пейджинг	53

4.5.3	Создание РКПД	54
4.5.4	Получение подразделения и связанный SevAssociation	54
4.6	Тесты REST API GraphQL	54
4.6.1	Требования	54
4.6.2	Состав пакета	55
4.6.3	Запуск тестов	55
4.6.4	Пример входа в систему	55
4.6.5	Примеры операций с элементом справочника	55
4.6.6	Примеры добавления РКПД	57
4.7	Дополнительная информация по пользовательским запросам	59
4.7.1	Пользовательские запросы GraphQL	59
4.7.2	Кастомные аргументы	60
5	C# Entity Framework (для ДЕЛЮ API 2023)	64
5.1	Модель данных, используемая в API	64
5.2	Внедрение зависимостей (DI)	69
5.3	Модули платформы	70
5.4	Контекст доступа к данным - IDataContext	70
5.4.1	Конвейер промежуточного ПО	72
5.5	Сервис DataContextHelperService	75
5.6	Права пользователя	78
5.6.1	Статический класс расширения UserExtensions	78
5.7	Сервисы кэширования	78
5.8	Пакетная модификация в базе данных	79
5.9	Работа с файлами	79
5.10	Логирование	80
6	Фоновые задачи	81
6.1	Введение в API фоновых задач	81
6.2	Модель данных, используемая в API фоновых задач	81
6.3	Разработка фоновой задачи (для ДЕЛЮ API 2009 и ДЕЛЮ API 2023)	81
6.3.1	Создание проекта	82
6.3.2	Создание класса фоновой задачи, наследника BaseEventDispatcher / BaseDispatcher / BaseDaemon	82
6.3.3	Чтение конфигурации фоновой задачи	88
6.3.4	Соединение с базой данных	91
6.3.5	Поиск объектов	91
6.3.6	Поиск ссылки на сущность системы	92

6.3.7	Загрузка объекта по идентификатор.....	92
6.3.8	Модификация данных объекта.....	92
6.3.9	Работа с email сообщениями	94
6.3.10	Использование спецхранилища значений (CUSTOM_STORAGE).....	95
6.3.11	Работа с событиями.....	96
6.4	Подготовка фоновой задачи к выполнению	98
6.4.1	Подготовка конфигурационного файла	98
6.4.2	Добавление metadata.json.....	111
6.4.3	Загрузка фоновой задачи в систему.....	111
6.4.4	Настройка фоновой задачи.....	112
6.5	Создание спец. событий для потребления ФЗ.....	114
6.6	Тестирование	117
7	Форматы обмена данными	121
7.1	Паспорт РК электронного сообщения E-mail	121
7.1.1	Перечень и содержание зон сообщения	121
7.1.2	Правила описания зон сообщения	121
7.1.3	Описание зоны "Заголовок"	122
7.1.4	Описание зоны "Документ"	122
7.1.5	Описание зоны "Расширение".....	125
7.1.6	Описание зоны "Уведомление".....	127
7.1.7	Описание элементов.....	127
8	Дополнительные функции (для ДЕЛО API 2009 и ДЕЛО API 2023).....	140
8.1	API входа в систему «ДЕЛО» и выхода из системы.....	140
8.2	API пользовательских настроек (USER_SETTINGS)	140
8.3	API системных настроек (APP_SETTINGS).....	141
8.4	Работа с временным хранилищем файлов FDULZ	142
8.4.1	Помещение файла во FDULZ.....	142
8.4.2	Использование содержимого файла из FDULZ в OData	142
8.4.3	Использование содержимого файла из FDULZ в GraphQL	142
8.4.4	Чтение файла из FDULZ	143
9	Разработка расширений rest api (для ДЕЛО API 2023).....	144
9.1	Пользовательские запросы	144
9.1.1	Запросы на чтение	144
9.1.2	Запросы на изменение.....	145
9.2	Контроллеры	145
10	Разработка отчётов	147

10.1.1	Создание и настройка модуля отчётов.....	147
10.1.2	Создание отчёта.....	148
10.1.3	Регистрация отчёта и команд	155
10.1.4	Переадресация	155
10.1.5	Подключение отчёта к стенду.....	156
11	Сборка проектов	158
Приложение А Использование API в реальных проектах		159
A.1	Импорт справочников из одного сервера системы «ДЕЛО» в другой.....	159
A.2	Веб-сервис создания РКПД по данным из кадровой и других систем.....	159
A.3	Плагин "Автоматизированная подготовка и отправка уведомлений заявителю"....	159
Приложение Б Описание интерфейсов и базовых классов ДЕЛО API 2009		161
B.1	Библиотека Eos.Delo.Common	161
	Описание элементов интерфейса IHead	161
	Описание элементов класса AppTrace.....	161
	Описание элементов класса EntityHelper	162
	Описание элементов класса Facade (Eos.Delo.App)	163
	Описание элементов класса LoadOptions.....	163
	Описание элементов класса WorkContext.....	163
	Описание элементов класса XmlSerializationHelper.....	164
	Описание элементов класса XmlValidator.....	165
	Другие элементы библиотеки.....	165
B.2	Библиотека Eos.Delo.Utills	165
	Описание элементов класса JsonConverter.....	165
	Описание элементов класса FileHelper.....	166
B.3	Библиотека Eos.Delo.Wapi.....	166
	Описание элементов класса MailManHelper	166
B.4	Библиотека Eos.Delo96.Api	166
	Описание элементов интерфейса IFileHelper (Eos.Delo96.Api.Entities.Helpers)	166
	Описание элементов класса ClassifHelper (Eos.Delo96.Api.Entities.Helpers).....	167
	Описание элементов класса ClassifEntity (Eos.Delo96.Api.Entities)	169
	Описание элементов класса ClassifTree (Eos.Delo96.Api.Entities).....	169
	Описание элементов класса CUSTOM_STORAGE_Helper (Eos.Delo96.Api.Entities.Helpers).....	170
	Описание элементов класса DOC_RC_Helper (Eos.Delo96.Api.Entities.Helpers)	171
	Описание элементов класса PRJ_RC_Helper (Eos.Delo96.Api.Entities.Helpers)	173
	Описание элементов класса RESOLUTION_Helper (Eos.Delo96.Api.Entities.Helpers)....	175

Описание элементов класса UserContext.....	176
Приложение В Системные настройки	177
В.1 Настройки системы.....	177
В.2 Тип объекта	177
В.3 Реквизиты	177
В.4 Типы данных системных настроек.....	178
В.5 Функции и права системных настроек	178
В.6 Примеры	178
В.7 Запись в AppSettings конкретного класса.....	179
В.8 Работа с Secret	180
В.9 Чтение и запись через API	182
Приложение Г Объекты системы	184
ACQUAINTANCE	184
ADDR_CATEGORY_CL	184
ADDRESS.....	185
ADDRESS_VID_CL.....	185
APP_HOST	186
APP_HOST_ASSIGNED_INSTANCE.....	186
APP_HOST_CONFIG	186
APP_HOST_CONFIG_DAEMON_TYPE.....	186
APP_HOST_INSTANCE.....	187
APP_HOST_PULSE	187
APP_HOST_VIRTUAL.....	187
APP_SETTINGS.....	188
APPDELTA.....	188
APPROACH.....	188
AR_CATEGORY.....	188
AR_CITIZEN_VALUE	189
AR_CLS	189
AR_CLS_CONTROL	189
AR_CLS_FIELD	190
AR_DESCRIPTOR.....	190
AR_DOCGROUP.....	191
AR_ORGANIZ_VALUE	192
AR_PRJ_VALUE.....	192
AR_RC_VALUE.....	192

AR_RUBRIC_VALUE	193
AR_VALUE_LIST.....	193
BANK_RECVISIT	193
BAR_CODE_SUPPORT	194
BPM_COMP_PROCESS_TYPE	194
BPM_COMPONENT_TYPE.....	194
BPM_INSTANCE.....	195
BPM_INSTANCE_COMP_ENDORSE	195
BPM_INSTANCE_COMP_EXEC	196
BPM_INSTANCE_COMP_PARAM	196
BPM_INSTANCE_COMP_PREP	196
BPM_INSTANCE_COMP_SIGN	197
BPM_INSTANCE_COMPONENT	197
BPM_PROC_VER_COMP_ENDORSE	198
BPM_PROC_VER_COMP_EXEC	198
BPM_PROC_VER_COMP_PARAM.....	198
BPM_PROC_VER_COMP_PREP	199
BPM_PROC_VER_COMP_SIGN.....	199
BPM_PROC_VER_COMPONENT	199
BPM_PROC_VER_DG	200
BPM_PROC_VER_OBJECT.....	200
BPM_PROC_VER_PRJ.....	200
BPM_PROC_VER_PRJ_STAGE.....	200
BPM_PROC_VER_SCHEDULE	201
BPM_PROCESS	201
BPM_PROCESS_ROLE.....	201
BPM_PROCESS_TYPE	202
BPM_PROCESS_VERSION	202
BPM_USER_PROCESS_TYPE	203
BUF_FOLDER.....	203
BUF_MESSAGE.....	204
BUF_MESSAGE_FIELDS	205
BUF_RECEIPT_COUNT	206
BUF_RULE.....	206
BUF_USER_FOLDER.....	206
BULK_DELETE_PROT.....	206

CA_CATEGORY	207
CABINET	207
CALENDAR_CL	207
CB_PRINT_INFO	208
CERTIFICATE	209
CITIZEN	209
CITIZEN_STATUS	211
CITSTATUS_CL	211
CL_SEARCH	212
CONNECTION_LOG	212
CONTACT	212
CUSTOM_STORAGE	213
DAEMON_GROUP	214
DAEMON_INSTANCE	214
DAEMON_INSTANCE_BIND	214
DAEMON_TYPE	215
DAEMON_TYPE_VERSION	215
DELIVERY_CL	215
DELO_BLOB	216
DELO_OWNER	216
DEP_REPLACE	216
DEPARTMENT	217
DEPARTMENT_REPL	219
DG_FILE_CATEGORY	219
DG_FILE_CONSTRAINT	219
DIADOC_EXCHANGE	219
DIADOC_SUBSCRIPTION	220
DOC_DEFAULT	220
DOC_DEFAULT_VALUE	220
DOC_EXE	221
DOC_FOLDER_ITEM	221
DOC_ORGANIZ_EXCLUDE	222
DOC_RC	222
DOC_SIGN	224
DOC_TEMPLATES	224
DOC_WHO	225

DOCGROUP_CL	225
DOCVID_CL	227
EDS_CATEGORY_CL	228
EOS_JOB	228
EOS_SCN	229
EVNT_FEED	229
EVNT_OBSERVED	230
EVNT_QUEUE_ITEM	230
EVNT_SUBSCRIPTION	231
EXT_PROT	231
EXT_PROT_OPER_TYPES	232
EXT_PROT_SETTINGS	232
FILE_CATEGORY_CL	232
FILE_CONTENTS	233
FILE_CONTENTS2	233
FILE_TYPE_CL	233
FOLDER	234
FORMAT_CL	234
FORUM_DUE	235
FORUM_MSG	235
FORUM_READ	235
FORUM_THEME	235
FORWARD	236
FUZZY_WORD	236
FUZZY_WORD_OBJ	237
HELPER_T1	237
JOURNAL	237
LIB_FILELINK	238
LIB_LIBRARY	238
LIB_PARAM	239
LINK_CL	239
LIST_ITEMS	240
MAIL_FOLDER	240
MAIL_RULE	240
MEDO_DISPATCHER_PROT	240
MEDO_NODE_CL	241

MEDO_PARTICIPANT	242
MESSAGES	242
MODULES.....	243
MREVT_ASSOCIATION	243
MREVT_EVENT	243
MREVT_REF.....	244
MTG_AR_MEETING_VALUE	245
MTG_DECISION_TYPE.....	245
MTG_INVITED	245
MTG_ISSUE_RUBRIC	246
MTG_ISSUE_TYPE	246
MTG_MEETING_RUBRIC	246
NEW_RECORD_CABINET	246
NOMENKL_CL	246
NOTIFY_CL	248
NTFY_EVENT.....	248
NTFY_MESSAGE	249
NTFY_OPERATION	249
NTFY_PERIODICITY.....	250
NTFY_SUBSCRIPTION	250
NTFY_SYSTEM_PARAMS	250
NTFY_USER_EMAIL.....	251
NTFYPF_CHANNEL	251
NTFYPF_MESSAGE.....	251
NTFYPF_MESSAGE_JOURNAL	252
NUMCREATION.....	252
OBJ_TEMPLATE	253
ORG_TYPE_CL.....	253
ORGANIZ_CL	254
PACK_ITEMS	255
PASS_STOP_LIST	255
PATCH_HISTORY.....	256
PATCH_HISTORY_CUSTOM.....	256
PLUGINS	256
PRINT_FILE_FORMAT	257
PRINT_FORM	257

PRINT_FORM_CATEGORY	258
PRJ_CURR_PRJ	258
PRJ_DEFAULT	258
PRJ_DEFAULT_VALUE	259
PRJ_ENDORSE_TASK	259
PRJ_EXEC	259
PRJ_FOLDER_ITEM	260
PRJ_FORUM	261
PRJ_NUMCREATION	261
PRJ_PREP_TASK	261
PRJ_RC	262
PRJ_REF_RUBRIC	264
PRJ_REF_SEND	264
PRJ_RETURN	265
PRJ_SIGN_TASK	265
PRJ_STAGE_CL	265
PRJ_VISA_SIGN	266
PROCESS_TASK	267
PROCESS_TASK_COMPETITOR	268
PROCESS_TASK_STATUS_CL	268
PROGRAMS	268
PROT	269
PROT_FILE_SSCAN	269
PROT_INFO	270
PROT_NAME	270
PROT_STREAM_SCAN	270
RAC_ADJUST	271
READ_PROT	272
REESTR_NEW	272
REESTRTYPE_CL	273
REF_ACCESS_CARD	274
REF_CONTEXT	274
REF_CORRESP	274
REF_FILE	275
REF_FILE_ACCESS	277
REF_FILE_EDS	277

REF_LETTER	278
REF_LINK	278
REF_RUBRIC	279
REF_SEND	280
REF_SOISP	281
REF_UFOLDER	281
REF_VISA	282
REGION_CL	282
REMINDER	283
REP_NUMCREATION	283
REP_NUMCREATION2	284
REPEAT_RES	284
REPLY	285
RESOLUTION	285
RESOLUTION_CARD	287
RESOLUTION_CATEGORY_CL	287
RESOLUTION_PRJ_COPY	288
RESPRJ_PRIORITY_CL	288
RESPRJ_STATUS_CL	289
RUBRIC_CL	289
SECURITY_CL	290
SEND_PACKAGE	291
SEV_ASSOCIATION	291
SEV_CHANNEL	292
SEV_COLLISION	292
SEV_PARTICIPANT	293
SEV_PARTICIPANT_RULE	293
SEV_REPORT	294
SEV_REPORT_EVENT	294
SEV_RULE	294
SEV_SUBSCRIBE	295
SEV_SYNC_REPORT	296
SHABLON_DETAIL	296
SIGN_KIND_CL	297
SMEV_PARTICIPANT	297
SMEV_SETTINGS_REQUEST	297

SMEV_SETTINGS_REQUEST_VERSION.....	298
SRCH_AR_HIER.....	298
SRCH_CATEGORY.....	298
SRCH_CRITERY.....	299
SRCH_GROUP.....	299
SRCH_GROUP_ITEM.....	300
SRCH_REQ_DESC.....	300
SRCH_REQUEST.....	300
SRCH_VIEW.....	301
SRCH_VIEW_DESC.....	302
SSCAN_BARCODE.....	302
SSCAN_REQUEST.....	302
SSTU_DECISION_HISTORY.....	303
STATUS_EXEC_CL.....	304
STATUS_REPLY_CL.....	304
STOP_WORDS.....	305
STTEXT.....	305
STTEXT_CONTROL.....	305
STTEXT_LIST.....	306
SW_LIST.....	306
SW_MODULES.....	306
SW_USE.....	307
T_WORDS.....	307
T_WU.....	307
TEMP_RC.....	307
UFOLDER.....	308
UFOLDER_ACCESS.....	308
USER_AUDIT.....	308
USER_AUTH_EXTERNAL_ID.....	309
USER_CABINET.....	309
USER_CARD_DOCGROUP.....	309
USER_CERT_PROFILE.....	310
USER_CERTIFICATE.....	310
USER_CL.....	310
USER_DOCGROUP_ACCESS.....	311
USER_EDIT_ORG_TYPE.....	312

USER_FILESECUR.....	312
USER_HISTORY.....	312
USER_LISTS	312
USER_ORGANIZ.....	313
USER_PARMS	313
USER_REQUEST.....	313
USER_RIGHT_DOCGROUP.....	314
USER_SETTINGS	314
USER_SRCH_GROUP.....	314
USER_TECH	314
USER_VIEW	315
USERCARD.....	315
USERDEP	315
USERSECUR.....	316
VIEWPROT	316
VISA_TYPE_CL.....	316
WAPI_SESSION.....	317
WAPI_SESSION_STORE	317
WEIGHT_DUE	317
Приложение Д Перечень связей таблиц системы «ДЕЛО».....	319
Приложение Е Специальные модификаторы данных	332
Е.1 Номерообразование	332
release_num_web – изменение счетчика номерообразования РК.....	332
release_prj_num_web – изменение счетчика номерообразования РКПД.....	332
Е.2 Протокол.....	332
write_prot – протоколирование изменения и просмотра объектов Дела.....	332
mark_read_prot – отметка прочтения пользователем объекта (РК, РКПД, поручения и т.д.).....	333
Е.3 События.....	333
add_evnt_queue_item – добавление элементов в очередь событий	333
save_custom_storage – сохранение данных в "произвольном" хранилище	334
Приложение Ж Нестандартные критерии поиска ДЕЛО API 2009	335
Приложение И Синтаксис запросов на текстовые поля ДЕЛО API 2009	340
Приложение К XSD-схемы файлов, используемых при организации электронного взаимодействия с системой «ДЕЛО»	341
К.1 Формат обмена по электронной почте.....	341
К.2 Формат обмена подсистемы СЭВ	341

Приложение Л Описание механизма событий БД «ДЕЛО»	343
Л.1 Возможности, предоставляемые механизмом событий	343
Л.2 Описание алгоритмов регистрации событий в очередях	343
Регистрация событий простых объектов	343
Регистрация событий сложных объектов	344
Регистрация событий сложных объектов с событием завершения сохранения	344
Приложение М Список имен объектов и их идентификаторов (KIND_OBJECT)	347
Приложение Н Права пользователя	354
Н.1 Статический класс расширения UserExtensions	354
Н.2 Класс UserRightsHelperService	354
Н.3 Класс PrjRcSecurityService	355
Н.4 Класс DocRcSecurityService	357
Н.5 Класс ResolutionSecurityService	357
Приложение П Сервисы кэширования	359
Приложение Р Сервис DataContextHelperService. Методы	360
Приложение С Перечень доступных UOD и их параметры (UOD_PROP) для составления отчетных форм	365

ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

«ДЕЛО»	система, автоматизирующая процессы документооборота организации
«ДЕЛО-WEB»	WEB-приложение, подсистема системы «ДЕЛО»
ДЕЛО API 2009	API системы «ДЕЛО», первая версия. Дата ввода в эксплуатацию: 2009 Актуальное состояние: используется.
ДЕЛО API 2023	API на ЭОС Платформе для системы «ДЕЛО», актуальная версия. Дата ввода в эксплуатацию: 2023 Актуальное состояние: внедрение.
Русские	
РК	регистрационная карточка документа
РКПД	регистрационная карточка проекта документа
РКС	регистрационная карточка События, содержащего данные мероприятия, экспертизы или собрания. В РКС можно добавить ссылки на РК/РКПД, с которыми связано данное событие
ДЛ	должностное лицо
АДЛ	ассоциированное должностное лицо
ЭП	электронная подпись
ИП	инициативные поручения
Каталог «Sdk» (название)	Каталог в дистрибутиве с примерами использования методов API, является частью поставки системы «ДЕЛО». В скобках - уточнение названия конкретного примера.
Английские	
API	Application Programming Interface: модуль, который позволяет программно (без пользовательского интерфейса) использовать все функции системы «ДЕЛО»
DI	Dependency Injections: один из основных принципов работы в современных приложениях на .NET Core. DI позволяет широко использовать собственные сервисы
Entity Framework Core	Технология для доступа к базам данных от Microsoft. Она позволяет взаимодействовать с СУБД с помощью сущностей (entity), то есть классов и объектов NET, а не таблиц базы данных.
FDULZ	FileDownloadUploadZone: временное хранилище файлов.
GraphQL	Язык запросов для API-интерфейсов и среда, в которой они выполняются. С помощью GraphQL можно получить данные из API и передать их в приложение (от сервера к клиенту).
SOp, COп	Service Operation, Сервисные Операции: вызов хранимой процедуры
OData	Open Data Protocol
ORM	Object-Relational Mapping
ISN	Identity Serial Number: уникальный номер

1 Архитектура системы «ДЕЛО»

1.1 Модель данных

Модель данных системы «ДЕЛО» представлена на Рисунк 1.

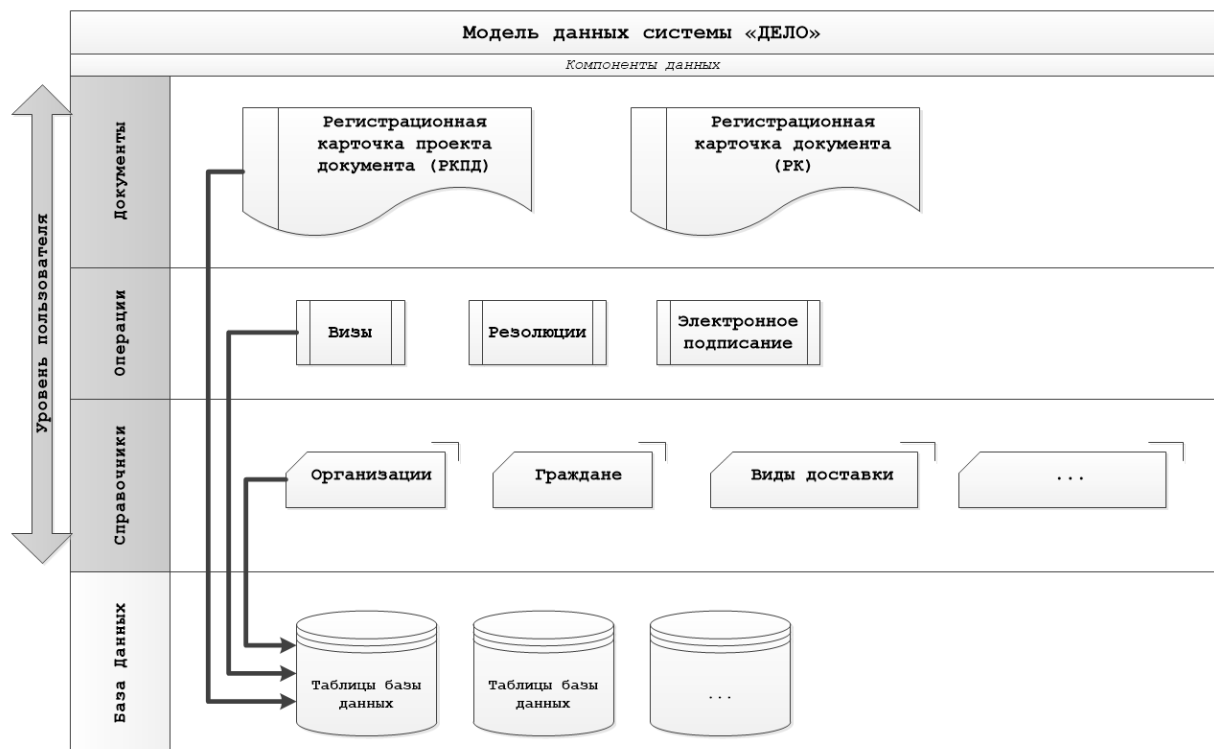


Рисунок 1

Прикладная объектная модель (карточки документов, визы, резолюции, и т.д.) описана в Приложение Г.

Все таблицы в системе «ДЕЛО» имеют следующие категории:

- Таблицы справочников (Группы документов, Подразделения, Организации, Рубрики, ...)
- Таблицы данных
 - 1) Хранение реквизитов объектов (РК, РКПД, поручения)
 - 2) Хранение данных о движении (жизни) объектов (кабинеты, вхождение в картотеки, реестры внешней отправки)
- Таблицы настроек и прав пользователей (Пользователи, Параметры пользователя, Абсолютные права, Картотекозависимые права)
- Вспомогательные таблицы (Автоматическое обновление ПО)

Справочники имеют типовую структуру, которая включает в себя:

- Первичный ключ – ISN_LCLASSIF
 - 1) ISN_NODE для иерархических справочников
 - 2) При ссылке на справочник используется имя справочника – ISN_DELIVERY
- Видимое пользователю содержимое – CLASSIF_NAME
 - 1) Альтернативный ключ
- Порядок записей в списках – WEIGHT
- Примечания – NOTE
 - 1) не отягощены никакой бизнес логикой

- Записи, защищенные от удаления – PROTECTED
 - 1) Их наличие критически важно для функционирования всей системы
- Записи скрытые и не доступные для дальнейшего использования – DELETED
 - 1) На данные записи уже есть ссылки из сущностей системы
 - 2) Использование данных записей не желательно
 - 3) Тем не менее в отчетах и поиске эти записи доступны

К основным справочникам системы «ДЕЛО» относятся:

- Группы документов
 - 1) Отвечает за первичную классификацию по иерархическим группам
 - 2) Определяет ряд основных свойств, таких как вид регистрационного номера, значения по умолчанию для реквизитов и правила регистрации РК
- Подразделения, картотеки и кабинеты
 - 1) Определяет иерархическую структуру организации с точки зрения документооборота – картотеки, подразделения, должностные лица (ДЛ)
 - 2) Определяет принадлежность ДЛ и подразделений картотекам
 - 3) Задаёт кабинеты, в которые должны приходить оповещения для должностных лиц
- Организации
 - 1) Определяет иерархическую структуру внешних контрагентов вплоть до представителей организаций
 - 2) Содержит контактную информацию – адреса, телефоны

Основными таблицами данных системы «ДЕЛО» являются:

- Регистрационные карточки документов (РК)
- Регистрационные карточки проектов документов (РКПД)
- Поручения
 - 1) Это «подчиненные» сущности, не существующие без РК документа, и их жизненный цикл зависит от жизненного цикла РК документа

Рассмотрим каждую составляющую модели данных подробно.

1.1.1 Справочник Группы документов

- Иерархический справочник
 - 1) Вершины являются всего лишь группировкой
 - 2) РК и РКПД создаются только для листов
- Первичная классификация РК
- Вид РК (входящие, письма, исходящие) - RC_TYPE
 - 1) Вершины могут иметь RC_TYPE = 0
- Шаблон формирования регистрационного номера – SHABLON
- Для исходящих – возможность создавать проекты документов – PRJ_NUM_FLAG
- В таблицах DOC_DEFAULT_VALUE и PRJ_DEFAULT_VALUE в виде пар «имя-значение» хранятся бизнес правила регистрации и редактирования РК, которые может задать технолог
- В таблицах AR_DESCRIPT, AR_DOCGROUP – описания дополнительных реквизитов и их доступность на той или иной группе документов

1.1.2 Справочник Подразделения, картотеки и кабинеты

- Иерархический справочник
 - 1) Вершины – подразделения
 - 2) Листья – должностные лица
 - 3) В зависимости от требований бизнес логики, в разных местах системы допустимо использовать только вершины, только листья, как листья, так и вершины
- Поля, имеющие значение только для должностного лица: SURNAME, DUTY, ISN_CABINET, POST_H
- DEPARTMENT_INDEX – индекс ДЛ или подразделения
 - 1) Участвует в формировании регистрационного номера РК
- Контактная информация: PHONE, PHONE_LOCAL, FAX, E_MAIL, NUM_CAB
- Справочник позволяет хранить историю
 - 1) START_DATE – дата создания подразделения, назначения должностного лица
 - 2) END_DATE – дата расформирования (переименования) подразделения, перевода должностного лица
 - 3) Раньше пользовались только признаком DELETED

1.1.3 Справочник Организации

- Организации – иерархический справочник
 - 1) Вершины – только для группировки
- Информация об организации: FULL_NAME, ZIPCODE, CITY, ADDRESS, ...
- Таблица CONTACT – представители организации
 - 1) Для любой организации существует запись соответствующая организации как таковой (ISN_CONTACT = ISN_ORGANIZ AND ORDERNUM = 0). На жаргоне – «нулевой контакт»
 - 2) В записи «нулевого контакта» хранится общая для организации контактная информация – телефон, факс, email и пр., другие поля не задействуются
 - 3) Остальные записи таблицы – представители организации, для них действительны все поля: подразделение, должность, фамилия представителя, телефон, факс, email и пр.

1.1.4 РК документа

- Главная таблица - DOC_RC
- Важные дочерние таблицы - REF_CORRESP, REF_LETTER, REF_RUBRIC, FORWARD, JOURNAL, REF_SEND, ...
- Второстепенные дочерние таблицы - REF_VISA, REF_SOISP, DOC_SIGN, DOC_WHO, ACQUAINTANCE, ...
- Вхождение в картотеки – REF_ACCESS_CARD
- Идентификатор РК – поле ISN_DOC
 - 1) Генерируется автоматически системой
 - 2) Не виден конечному пользователю
- Вид РК – поле KIND_DOC

- 1) Входящие = 1, письма = 2, исходящие = 3
- 2) Избыточное – дублирует DUE_DOCGROUP, для быстрого отсева «ненужных» при запросе
- 3) Никогда не меняется
- Пользовательский идентификатор PK – поля DUE_DOCGROUP + DOC_DATE+ FREE_NUM
 - 1) Группа документов, дата регистрации и регистрационный номер
 - 2) Не поддерживается констрейнтами БД – только «силами» приложений
- Гриф доступа – SECURLEVEL
 - 1) В паре с DUE_DOCGROUP влияет на права пользователей на чтение PK
- Картотека и кабинет регистрации – ISN_CARD_REG и ISN_CABINET_REG
 - 1) Картотека – обязательна, кабинет – нет
- Блок контрольности PK – CONTROL_STATE, PLAN_DATE, FACT_DATE, DELTA_DATE
 - 1) Ведет себя по-разному в зависимости от значения параметра «правило определения контрольности документа»
 - 2) CONTROL_STATE - Не контрольные = null, На контроле = 1, Снято с контроля = 2
 - 3) **Внимание:** в запросах нельзя писать CONTROL_STATE <> 2 нужно NVL(CONTROL_STATE, 0) <> 2 или NOT (CONTROL_STATE = 2)

1.1.5 Поручения

- Главная таблица – RESOLUTION
- Дочерняя – Исполнители поручений – REPLY
- Напоминания - REMINDER
- Поручения (резолуции и пункты)
 - Идентификатор поручения – ISN_RESOLUTION
 - Вид поручения – RESOLUTION_KIND
 - 1) Резолюция = 1, Пункт = 2
 - Текст поручения – RESOLUTION_TEXT
 - Номер пункта – ITEM_NUMBER
 - Автор резолюции – DUE
 - 1) Действительно только для резолюций
 - Дата рассылки проекта резолюции SEND_DATE
 - 2) Является признаком «Проекта резолюции» – SEND_DATE IS NULL
 - 3) Для пунктов = дате регистрации документа
 - Блок контрольности – CONTROL_STATE, PLAN_DATE, FACT_DATE, DELTA_DATE
 - 1) Аналогично контрольности PK
 - Контролер поручения – DUE_CONTROLLER
 - 1) Может быть заполнено в том числе и для неконтрольных поручений
 - Организация списка поручений – ISN_PARENT_RESOLUTION, LAYER, WEIGHT
 - 1) Не бывает подчиненных пунктов
 - 2) ISN_PARENT_RESOLUTION IS NULL – поручения первого уровня
- Поручения (исполнители)

- Список исполнителей поручения – REPLY
 - 1) Эта же таблица хранит данные отчетов об исполнении поручения
- Ссылка на поручение – ISN_RESOLUTION
- Должностное лицо – исполнитель - DUE
- Ответственные исполнители – отмечены «флагом» - MAIN_FLAG
 - 1) Ответственных может быть несколько, но чаще всего один
- Сроки исполнения – PLAN_DATE, REPLY_DATE, DELTA_DATE
 - 1) Плановый срок берется из поручения
 - 2) Фактический срок – отвечает за признак «исполненности»
 - 3) Разница – позволяет быстро найти «просроченных» исполнителей
- Текст отчета – REPLY_TEXT
 - 1) Отчет может быть в виде прикрепленного файла

1.2 Функциональная схема

Функциональная схема «ДЕЛО» представлена на Рисунок 2.

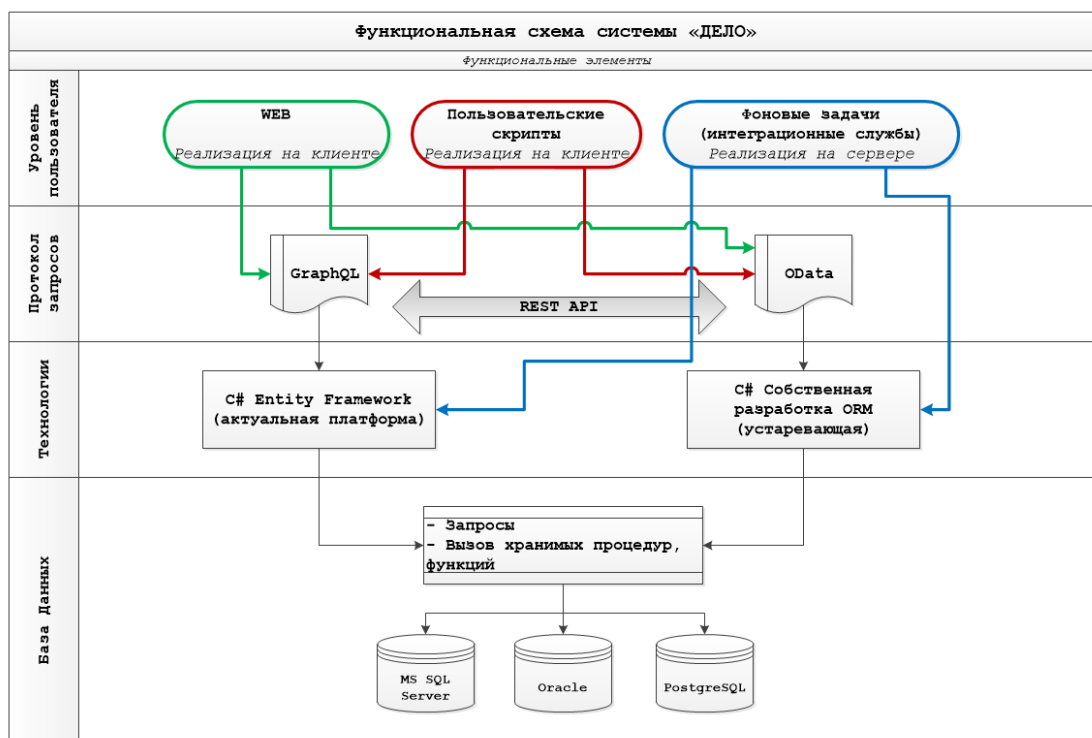


Рисунок 2

На уровне пользователя API дает возможность без использования интерфейса реализовывать основные функции системы «ДЕЛО». Как на клиенте (пользовательские скрипты), так и на сервере приложений (фоновые задачи). Подробно о серверной реализации фоновых задач см. 6. Фоновые задачи.

1.3 Технологические стеки

В системе «ДЕЛО» существует 2 стека технологий, разработанные в разное время:

- ДЕЛО API 2009 (начало эксплуатации – 2009 год)
- ДЕЛО API 2023 (начало эксплуатации – 2023 год)

В приложениях на момент написания данной документации могут использоваться одновременно возможности обоих стеков. Об окончательном переходе на стек

ДЕЛО API 2023 и вывода из эксплуатации стека ДЕЛО API 2009 клиенты будут оповещены дополнительно. Оповещение будет опубликовано на официальном сайте компании в разделе «Файлы и обновления» (https://eos.ru/eos_support/eos_downloads/fajly-i-obnovleniya-delo/) как минимум за 6 месяцев до выпуска версии, в которой будет отключена поддержка ДЕЛО API 2009.

1.3.1 ДЕЛО API 2009

Структура стека показана на Рисунок 3.



Рисунок 3

Для доступа к базам данных используется собственная разработка ORM (object-relational mapping) – C# ORM ЭОС.

В качестве инструмента запросов и обновления данных по HTTP протоколу используется OData (Open Data Protocol).

1.3.2 ДЕЛО API 2023

Структура стека показана на Рисунок 4.

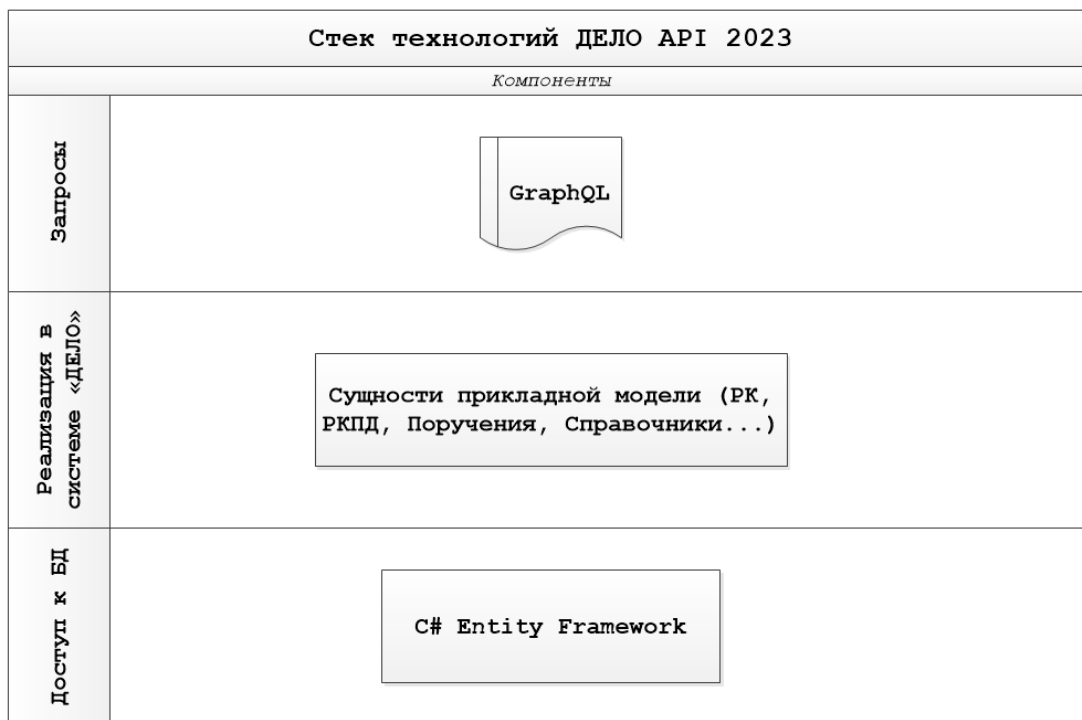


Рисунок 4

В стеке ДЕЛО API 2023 для доступа к базам данных используется Entity Framework Core от Microsoft. Это позволяет взаимодействовать с СУБД с помощью сущностей (entity), то есть классов и объектов NET, а не таблиц базы данных. На сегодня - это самый распространенный и функциональный ORM-инструмент в C# для отображения данных на реальные объекты.

В качестве инструмента обработки запросов и манипулирования данными используется GraphQL.

2 OData (для ДЕЛО API 2009)

2.1 Описание протокола

В качестве протокола обмена информацией используется протокол HTTP.

Используемые методы протокола HTTP: GET, POST.

Обработку запросов осуществляет сервис предоставляющий доступ к API и работающий на платформе EOS Platform Host.

Допустим, сервис зарегистрирован по адресу <https://delo.server.org/> (далее <адрес сервиса>).

Все обращения к API осуществляются через <адрес сервиса> >/CoreHost/OData/<путь и параметры запроса>.

Результат запроса (если не определено иное) будет представлен в виде объекта или массива объектов в формате JSON в теле ответа с Content-Type:application/json;charset=UTF-8.

В случае ошибки в возвращаемом объекте присутствует свойство «OData.error», содержащее объект, описывающий произошедшую ошибку.

Создание сессии и закрытие сессии в системе «ДЕЛО» являются универсальными для ДЕЛО API 2009 (OData) и ДЕЛО API 2023 (GraphQL).

Описание и примеры кода см. 8.1. API входа в систему «ДЕЛО» и выхода из системы.

2.1.1 Идентификатор объекта

Может быть представлен в зависимости от типа объекта как:

- **Число** – целое число.
- **Строка** – строка в одинарных кавычках, при иерархической структуре содержит точки как разделители в описании пути к объекту от родительской вершины, например ‘0.’ – корневая вершина, ‘0.2SD.’ – объект, входящий в корневую вершину, ‘0.2SD.2SP’ – объект входящий в упомянутый объект и т.д.

2.1.2 Временный идентификатор объекта

Используется при добавлении объектов в рамках пакетной команды. Должен быть уникальным для каждого добавляемого объекта в рамках пакета команд.

Представлен отрицательным числом, которое меньше или равно -10000. Например: -10100, -10101 и т.д.

Если используется строковый идентификатор при иерархической структуре, используется отрицательное число внутри строки. Например: ‘0.2SD.-10000’ для нового объекта, входящего в объект с идентификатором ‘0.2SD.’.

2.1.3 Представление результата

Результат запроса (если не определено иное) будет представлен в виде объекта в формате JSON в теле ответа с Content-Type:application/json;charset=UTF-8. В объекте обычно присутствует поле «OData.metadata», содержащее информацию о виде возвращаемого объекта. Если это выборка единичного объекта, то объект будет содержать содержимое объекта. Когда осуществляется выборка потенциально нескольких объектов, будет содержаться поле «value», содержащее объект или массив объектов в формате JSON. Если содержится массив, то также присутствует поле «\$inlinecount» содержащее целое число количества записей в массиве.

Объект СЭД содержит следующие типы сущностей (см. пункты настоящего подраздела):

Значения полей

Значения скалярных полей (строка, число и т.д.), например:

...

```
"ADDRESS_FLAG":0,
"ANNOTAT":"О направлении регламента",
"CONTROL_STATE":null,
"DOC_DATE":"2012-04-27T00:00:00",
...
```

Списки вложенных объектов

Представляют собой массив сериализованных вложенных объектов. Имя списка всегда заканчивается на `_List`. Например:

```
...
"REF_ACCESS_CARD_List":[
  ISN_REF_DOC":4353,
  "DUE":"0.",
  "KIND_DOC":3,
  "DATE_CARD":"2012-04-27T12:56:46",
  "CONTROL_STATE":null,
  ...],
  {
    "ISN_REF_DOC":4353,
    "DUE":"0.2SF.2T7.",
    ...
  }],
"REF_CORRESP_List":[
],
...
```

2.1.4 Опции сериализации (\$expand)

Опции указываются как параметры запроса и определяют данные, которые будут сериализованы.

- **\$expand** – определяет, включать или нет списки вложенных объектов. Если опция не указана, списки в сериализацию не включаются.

2.1.5 Указание картотеки и кабинета

При операциях с РК имеет значение, в рамках какой картотеки и кабинета происходит операция. Идентификаторы картотеки и кабинета указываются в параметрах запроса.

- **card_id** – идентификатор картотеки.
- **cabinet_id** – идентификатор кабинета.

2.1.6 Параметр criteries

В этом параметре содержатся критерии выборки объектов из базы данных. В зависимости от типа запроса эти данные могут располагаться:

- Только параметрах запроса GET.

Параметр представляет собой объект в формате JSON, содержащий критерии запроса и выражение для поиска по соответствующему критерию.

Формат:

```
{ "Имя критерия": "Запрос" [ "Имя критерия": "Запрос" [, ...] ] }
```

При использовании критериев выборки в ответе появляется свойство «\$TotalRecords», содержащее целое число всех записей, которые найдены с использованием данных критериев.

Например, необходимо найти документ с номером Л-1. Параметр `criteries` будет выглядеть так:

```
{ "Rc.RegNum": "Л-1" }
```

С учетом URL encode запрос будет таким:

```
<адрес сервера>/CoreHost/OData/DOC_RC?criteries=%7B"Rc.Regnum"%3A"Л-1"%7D
```

Соответственно результат:

```
{ "OData.metadata": ".../$metadata#DOC_RC", "$inlinecount": 1, "$TotalRecords": 1, "value": [ { "ISN_DOC": 4363, "KIND_DOC": 3, "DUE_DOCGROUP": "0.2U9.2UX.", "DOC_DATE": "2012-04-27T00:00:00", "ORDER_NUM": 0, "FREE_NUM": "Л-1", "ANNOTAT": "Об аренде", "CONSISTS": "6", "SPECIMEN": "1", "DUE_CARD_REG": "0." ... } ] }
```

2.1.7 Пакетные команды

Пакетные команды позволяют выполнять за один запрос несколько команд.

Пакетная команда использует метод HTTP POST. В теле запроса содержится пакет в виде текста. Принимается только запрос с Content-Type: application/x-www-form-urlencoded; charset=UTF-8.

Адрес для вызова пакетной команды формируется как: <адрес сервиса>/CoreHost/OData/\$changes.

В теле запроса отправляется массив объектов в формате JSON.

Формат пакета команд

Пакет команд состоит из объектов со следующими свойствами:

method – команда (POST, DELETE или MERGE).

url – ресурс: путь к объекту или вызов сервисного метода с параметрами, аналогичен строке запроса, которая формируется для GET запроса, без адреса сервиса и ./CoreHost/OData/.

data – данные объекта.

Виды команды:

- **POST** – создает объект с данными, указанными в параметре data;
- **DELETE** – удаляет указанный объект;
- **MERGE** – изменяет указанный объект согласно данным в параметре data.

Путь к объекту содержит последовательные наименование типа объекта или списка объектов со значением идентификатора в круглых скобках, если он уже существует, и слэш («/») в качестве разделителя уровня вложенности. Для добавляемого объекта его идентификатор в пути не указывается.

Например:

```
[ { "method": "MERGE", "url": "DOC_RC(4568)", "data": { "NOTE": "3rv3v3rv3vrv3v3v" } }, { "method": "DELETE", "url": "DOC_RC(4568)/DOC_WHO_List(4569)" }, { "method": "POST", "url": "DOC_RC(4568)/DOC_WHO_List", "data": { "ISN_DOC_WHO": -19999, "ISN_DOC": 4568, "ORDERNUM": 2, "DUE_PERSON": "0.2SD.2SR." } } ]
```

Результат пакетной команды

Результат пакетной команды возвращается в виде JSON массива, где каждый элемент по порядку соответствует команде в пакете. В каждом присутствует значение statusCode, которое соответствует коду состояния HTTP (201 – «создано», 204 – «нет содержимого» и т.д.). В свойстве data, если таковое есть, содержится дополнительная информация.

При создании объекта возвращается только его идентификатор. Содержимое нового объекта получается командой чтения данных.

Например:

```
[ { "statusCode": 204 }, { "statusCode": 204 }, { "statusCode": 201, "data": { "OData.metadata": ".../$metadata#MapItemInt", "ID": 4574, "TempID": -19999 } } ]
```

2.1.8 Вызов сервисных методов

Сервисные методы - это функциональность API, которая включает в себя бизнес-логику, которую нецелесообразно реализовывать через доступ к объектам базы данных.

Некоторые методы вызываются методом GET.

Формат <адрес сервиса>/CoreHost/OData/<имя сервисного метода>[?параметры].

Например, вызов сервисного метода создания заготовки для РКПД (здесь пример раскодирован из UriEncode):

```
<адрес сервиса>/CoreHost/OData/API_Build_PRJ_RC?regParams={"DOC_DATE":"2021-12-16T17:34:02.3899466+03:00","DUE_DOCGROUP":"0.2UB.2UD.2UJ."}&$expand=REF_FILE_List/REF_FILE_EDS_List,REF_FILE_List/REF_FILE_ACCESS_List,PRJ_VISA_SIGN_List,PRJ_EXEC_List,PRJ_REF_SEND_List,PRJ_REF_RUBRIC_List,PRJ_VERSION_List,REF_LINK_List,AR_PRJ_VALUE_List
```

Список параметров, их ожидаемый тип и результат, возвращаемый сервисным методом, индивидуален для каждого сервисного метода.

Также некоторые сервисные методы можно вызывать в рамках пакетной команды, и возвращается результат также в рамках возвращаемых значений из пакетной команды. Вид команды в таком случае следует указать «POST».

2.2 Функции API

2.2.1 Загрузка объектов по списку идентификаторов

Выполняется путем вызова запроса:

<адрес сервиса>/CoreHost/OData/<имя объекта>(<идентификатор>)

ИЛИ

<адрес сервиса>/CoreHost/OData/<имя объекта>?ids=<список идентификаторов>.

Возможно использовать Опции сериализации.

Например, при необходимости загрузить объекты типа DELIVERY_CL с идентификаторами 1, 2, 3 и 4, не включая ссылки и списки. Запрос:

```
<адрес сервиса>/CoreHost/OData/DELIVERY_CL?ids=1%2c2%2c3%2c4
```

Результат:

```
{
  "OData.metadata": ".../$metadata#DELIVERY_CL", "$inlinecount": 4, "value":
  [{"ISN_LCLASSIF": 1, "WEIGHT": 3774,
    "CLASSIF_NAME": "E-MAIL",
    "PROTECTED": 1, "DELETED": 0, "NOTE": null},
   {"ISN_LCLASSIF": 2,
    ...},
   {"ISN_LCLASSIF": 3,
    ...},
   {"ISN_LCLASSIF": 4,
    ... } ] }
```

2.2.2 Поиск объектов

Выполняется путем вызова запроса <адрес сервиса>/CoreHost/OData/<имя объекта>.

Критерии запроса указываются в Параметре criteries.

Возможно использовать Опции сериализации.

Например, необходимо найти объект DOC_RC с номером Ф-3. Запрос:

```
<адрес сервиса>/CoreHost/OData/DOC_RC?criteries=%7b%22Rc.RegNum%22%3a%22D0%A4-3%22%7d
```

Результат:

```
{ "OData.metadata": ".../$metadata#DOC_RC", "$inlinecount": 1, "$TotalRecords": 1,
  "value": [{"ISN_DOC": 4333, "KIND_DOC": 1, "DUE_DOCGROUP": "0.2TZ.2U5.", "DOC_DATE": "2012-04-27T00:00:00", "ORDER_NUM": 3, "FREE_NUM": "Ф-3", "ANNOTAT": "О направлении регламента", "CONSISTS": "2", "SPECIMEN": "1", "DUE_CARD_REG": "0.", "ISN_CABINET_REG": 4089, "ISN_NOMENC": null, "SECURLEVEL": 1, "ISN_DELIVERY": 3774, "ISREPEAT": null, "ISCOLLECTIV
```

```
E":null,"ADRESS_FLAG":0,"TEL_NUM":null,"ANONIM":null,"DUE_PERSON_EXE":null,"CONTROL_STATE":null,"PLAN_DATE":"2012-05-18T00:00:00","FACT_DATE":null,"NOTE":null,"FILE_COUNT_VISIBLE":0,"NOTHARDCOPY":0,"CITO":0,"E_DOCUMENT":0,"ACCESS_MODE":0,"INS_WHO":null,"INS_DATE":null,"OTHER_WHO":1,"FREE_NUM_SORT":null,"SENDER_NUM":null,"EDIT_REASON":null,"ISN_DOCVID":0}}}
```

Рассмотрим на конкретных примерах правила формирования критериев поиска:

Пример 1

```
$Response = $null
$query = "{`\"CLASSIF_NAME`:\"=МЭДО`\"}"
$Response = Invoke-WebRequest -Uri "<адрес сервера>/CoreHost/OData/DELIVERY_CL?criteriaes=$( [uri]::EscapeDataString($query) )" -Method Get -WebSession $Session -ContentType "application/json"
$Response.Content | ConvertFrom-Json | ConvertTo-Json -Depth 10
```

Пример 2

```
$Response = $null
$query = "{`\"DOC_RC.REF_CORRESP.ORGANIZ.CLASSIF_NAME`:\"`финанс`\"}"
$Response = Invoke-WebRequest -Uri "<адрес сервера>/CoreHost/OData/DOC_RC?criteriaes=$( [uri]::EscapeDataString($query) )" -Method Get -WebSession $Session -ContentType "application/json"
$Response.Content | ConvertFrom-Json | ConvertTo-Json -Depth 10
```

Пример 3

```
$Response = $null
$query = "{`\"DOC_RC.DOC_EXE.PERSON.CLASSIF_NAME`:\"`рубан`\"}"
$Response = Invoke-WebRequest -Uri "<адрес сервера>/CoreHost/OData/DOC_RC?criteriaes=$( [uri]::EscapeDataString($query) )" -Method Get -WebSession $Session -ContentType "application/json"
$Response.Content | ConvertFrom-Json | ConvertTo-Json -Depth 10
```

Пример 4

```
$Response = $null
$query = "{`\"DOC_RC.ISN_NOMENC`:4057876`\"}"
$Response = Invoke-WebRequest -Uri "<адрес сервера>/CoreHost/OData/DOC_RC?criteriaes=$( [uri]::EscapeDataString($query) )" -Method Get -WebSession $Session -ContentType "application/json"
$Response.Content | ConvertFrom-Json | ConvertTo-Json -Depth 10
```

Пример 5

```
$Response = $null
$query = "{`\"DOC_RC.NOMENC.CLASSIF_NAME`:\"`записки`\"}"
$Response = Invoke-WebRequest -Uri "<адрес сервера>/CoreHost/OData/DOC_RC?criteriaes=$( [uri]::EscapeDataString($query) )" -Method Get -WebSession $Session -ContentType "application/json"
$Response.Content | ConvertFrom-Json | ConvertTo-Json -Depth 10
```

Пример 1: в таблице DELIVERY_CL (справочник видов доставки) ищем записи, у которых поле CLASSIF_NAME (наименование видов доставки) точно соответствует МЭДО. При этом перед наименованием поля указывать наименование таблицы не обязательно.

Пример 2: в таблице DOC_RC (ПК документов) ищем все записи, полученные из организаций, которые содержат в своем названии «финанс» (Министерство финансов, Центр финансирования и т.д.)

В самом запросе префикс DOC_RC не обязателен.

Пример 2

```
...
$query = "{`"[DOC_RC].REF_CORRESP.ORGANIZ.CLASSIF_NAME`":`"финанс`"}"
...
```

В таблице REF_CORRESP (Корреспонденты и Сопроводительные документов РК) ссылкой на справочник ORGANIZ_CL (Организаций) является поле DUE_ORGANIZ. В критерии поиска ссылка представлена именем поля ORGANIZ с исключенным префиксом DUE_.

Общее правило: В запросах для полей, содержащих префиксы DUE_, ISN_ эти префиксы отбрасываются.

Пример 3: в таблице DOC_RC (ПК документов) ищем все записи, фамилии исполнителей которых содержат «рубан» (Рубанский, Зарубанов и т.д.)

DOC_EXE таблица исполнителей, в которой есть поле DUE_PERSON. По Перечню связей таблиц (Приложение Д) это ссылка в таблицу DEPARTMENT (Справочник подразделений (ДЛ)), из которой мы указываем в критерии поиска CLASSIF_NAME.

Общее правило: В запросах для полей, содержащих префиксы DUE_, ISN_ эти префиксы отбрасываются.

Примеры 4-5: по сути один и тот же запрос, сформулированный разными способами.

Пример 4: в таблице DOC_RC (ПК документов) ищем все записи, с идентификатором ISN_NOMENC (Номенклатура дел) равным 4057876.

Обратите внимание, что название поля указано полностью ISN_NOMENC, префикс ISN сохранен.

Пример 5: в таблице DOC_RC (ПК документов) ищем все записи, в названии Номенклатуры дел которых есть «записки» (идентификатор 4057876 в справочнике Номенклатуры дел NOMENC_CL).

В DOC_RC ссылкой на справочник NOMENC_CL (Номенклатуры дел) является поле ISN_NOMENC. В критерии поиска ссылка представлена именем поля NOMENC с исключенным префиксом ISN_.

Общее правило: В запросах для полей, содержащих префиксы DUE_, ISN_ эти префиксы отбрасываются.

Для случаев нестандартных критериев поиска разработаны специальные SQL-запросы, которые могут вызываться пользователем. Их описание см. в Приложение Ж.

2.2.3 Модификация объектов

Производится с помощью пакетной команды **MERGE**.

В поле «url» приводится путь к объекту от родительского с указанием идентификатора

Значения передаются в параметре data.

Пример:

```
[{
  "method": "MERGE",
  "url": "DOC_RC(4451) ",
  "data": {
    ...
    "ANNOTAT": "Новая аннотация",
    ...
  }
}]
```

2.2.4 Создание объектов

Производится с помощью пакетной команды **POST**.

Значения передаются в параметре data.

Пример:

```
[{
  "method": "POST",
  "url": "DELIVERY_CL",
  "data": {
    "ISN_LCLASSIF": -10000,
    "CLASSIF_NAME": "тест"
  }
}]
```

2.2.5 Удаление объектов

Производится с помощью пакетной команды **DELETE**.

Команда не содержит дополнительных параметров.

Пример:

```
[{
  "method": "DELETE",
  "url": "DELIVERY_CL(4058451)"
}]
```

2.2.6 Получение содержимого файла

Для ДЕЛО API 2023: Выполняется путем вызова запроса <адрес сервиса>/CoreHost/FOP/GetFile/<isn файла>/<имя файла(произвольное)>.

Возвращает содержимое файла в теле ответа. Заголовок Content-Type выставляется согласно расширению файла. Заголовок Content-Disposition:attachment содержит атрибут filename в соответствии с данными из системы.

2.2.7 Создание шаблона для новой РК

Производится при помощи запуска сервисной операции: <адрес сервиса>/CoreHost/OData/API_Build_DOC_RC?regParams=<параметры регистрации>&card_id=<ID картотеки, в которой регистрируется РК>&\$expand=<список вложенных объектов>.

Первый аргумент представляет собой строку с объектом в JSON формате с полями DUE_DOCGROUP, содержащим ID группы документов создаваемой РК, и DOC_DATE, содержащее дату регистрации. Параметр \$expand заполняется в соответствии данной опции сериализации и содержит список вложенных объектов.

Возвращает содержимое, аналогичное возвращаемому при загрузке объектов, но для ещё не созданной РК с временными ID, заполненными по умолчанию полями и сгенерированным номером, если в группе документов есть номеробразование.

Например:

```
<адрес сервиса>/CoreHost/OData/
API_Build_DOC_RC?regParams=%27%7B%22DUE_DOCGROUP%22%3A%220.2U9.2UR.%22%2C%22DOC_DATE%22%3A%222021-03-08%22%7D%27&card_id=0.&$expand=REF_ACCESS_CARD_List,REF_FILE_List,REF_FILE_List/REF_FILE_EDS_List,REF_FILE_List/REF_FILE_ACCESS_List,DOC_SIGN_List,DOC_EXE_List,REF_VISA_List,DOC_WHO_List,REF_SEND_List,REF_SOISP_List,REF_LETTER_List,REF_RUBRIC_List,REF_RUBRIC_List/AR_RUBRIC_VALUE_List,REF_CORRESP_List,REF_LINK_List,JOURNAL_List,AR_RC_VALUE_List,FORWARD_List
```

```
{ "OData.metadata": "../$metadata#DOC_RC", "$inlinecount": 1, "value": [ { "ISN_DOC": -10000, "KIND_DOC": 3, "DUE_DOCGROUP": "0.2U9.2UR.", "DOC_DATE": "2021-03-08T00:00:00", "ORDER_NUM": 2, "FREE_NUM": "-2", "ANNOTAT": null, "CONSISTS": null, "SPECIMEN": "1", "DUE_CARD_REG": "0.", "ISN_CABINET_REG": 4089, "ISN_NOMENC": null, "SECURLEVEL": 1, "ISN_DELIVERY": null, "ISREPEAT": null, "ISCOLLECTIVE": null, "ADDRESS_FLAG": 0, "TEL_NUM": null, "ANONIM": null, "DUE_PERSON_EXE": null, "CONTROL_STATE": null, "PLAN_DATE": null, "FACT_DATE": null, "NOTE": null, "FILE_COUNT_VISIBLE": 0, "NOTHARDCOPY": 0, "CITO": 0, "E_DOCUMENT": 0, "ACCESS_MODE": 0, "INS_WHO": null, "INS_DATE": null, "OTHER_WHO": null, "FREE_NUM_SORT": null, "SENDER_NUM": null, "EDIT_REASON": null, "ISN_DOCVID": 0, "REF_ACCESS_CARD_List": [], "REF_
```

```
FILE_List":[],"DOC_SIGN_List":[],"DOC_EXE_List":[],"REF_VISA_List":[],"DOC_WHO_List":[],"REF_SEND_List":[],"REF_SOISP_List":[],"REF_LETTER_List":[],"REF_RUBRIC_List":[],"REF_CORRESP_List":[],"REF_LINK_List":[],"JOURNAL_List":[],"AR_RC_VALUE_List":[{"ISN_RC":-10000,"eos_item_number":null,"eos_item_text":null,"EOS_GUID_RC":null,"end_dog":null,"type_dog":null,"sum_dog":null,"kontragent":null,"date_end":null,"predmet":null,"eos_ir_status":null,"eos_ir_text":null,"eos_ir_date":null,"eos_ir_link":null,"eos_ir_notification":null,"eos_ir_paper_reply":null,"eos_ir_e_reply":null,"eos_ir_e_mail":null,"otmetka_oznakomlenija":null,"EOS_PUBLISHED_NUMBER":null,"EOS_PUBLISHED_DATE":null,"EOS_PUBLISHED_POINT":null,"EOS_PUBLISHED_CONTROL":null,"eos_item_principal":null,"eos_ir_sending":null,"MEDO_DOC_KIND":null,"bgfdbfd":null}],{"FORWARD_List":[]}]}
```

2.2.8 Создание шаблона для нового РКПД

Производится при помощи запуска сервисной операции: <адрес сервиса>/CoreHost/OData/API_Build_PRJ_RC?regParams=<параметры регистрации>&\$expand=<список вложенных объектов>.

Первый аргумент представляет собой строку с объектом в JSON формате с полями DUE_DOCGROUP, содержащим ID группы документов создаваемой РКПД, и DOC_DATE, содержащее дату регистрации. Параметр \$expand заполняется в соответствии данной опции сериализации и содержит список вложенных объектов.

Возвращает содержимое, аналогичное возвращаемому при загрузке объектов, но для ещё не созданной РКПД с временными ID, заполненными по умолчанию полями и сгенерированным номером, если в группе документов есть номеробразование.

Например:

```
<адрес сервиса>/CoreHost/OData/API_Build_PRJ_RC?regParams=%27%7B"DUE_DOCGROUP"%3A"0.2U9.2UX."%2C"DOC_DATE"%3A"2021-03-08"%7D%27&$expand=REF_FILE_List/REF_FILE_EDS_List,REF_FILE_List/REF_FILE_ACCESS_List,PRJ_VISA_SIGN_List,PRJ_EXEC_List,PRJ_REF_SEND_List,PRJ_REF_RUBRIC_List,PRJ_VERSION_List,REF_LINK_List,AR_PRJ_VALUE_List
```

Ответ:

```
{"OData.metadata": "../$metadata#PRJ_RC", "$inlinecount": 1, "value": [{"ISN_PRJ": -10000, "ISN_PRJ_BATCH": -10000, "PRJ_VERSION": 1, "CURRENT_FLAG": 1, "PRJ_STAGE": 1, "DUE_DOCGROUP": "0.2U9.2UX.", "SECURLEVEL": 1, "FREE_NUM": "3", "PRJ_DATE": "2022-03-08T00:00:00", "ORDER_NUM": 3, "PLAN_DATE": null, "ANNOTAT": null, "CONSISTS": null, "NOTE": null, "AUTO_REG": 0, "ISN_NOMENKL": null, "APPLY_EDS": null, "APPLY2_EDS": null, "APPLY_EXEC_EDS": 0, "DEL_AFTER_REG": null, "FILE_COUNT": 0, "FILE_COUNT_VISIBLE": 0, "E_DOCUMENT": 0, "ISN_DOCVID": 0, "REF_FILE_List": [], "PRJ_VISA_SIGN_List": [], "PRJ_EXEC_List": [{"ISN_PRJ_EXEC": -10001, "ISN_PRJ": -10000, "DUE_PERSON": "0.2SV.2SX.", "ORDERNUM": 1, "ADDINFO": "Автор", "CAN_MANAGE_EXEC": 1, "CAN_WORK_WITH_PRJ": 1, "CAN_WORK_WITH_FILES": 1, "CAN_MANAGE_APPROVAL": 1, "FOUNDER_FLAG": 1}], "PRJ_REF_SEND_List": [], "PRJ_REF_RUBRIC_List": [], "PRJ_VERSION_List": [], "REF_LINK_List": [], "AR_PRJ_VALUE_List": [{"ISN_PRJ": -10000, "eos_item_number": null, "eos_item_text": null, "EOS_GUID_RC": null, "end_dog": null, "type_dog": null, "sum_dog": null, "kontragent": null, "date_end": null, "predmet": null, "eos_ir_status": null, "eos_ir_text": null, "eos_ir_date": null, "eos_ir_link": null, "eos_ir_notification": null, "eos_ir_paper_reply": null, "eos_ir_e_reply": null, "eos_ir_e_mail": null, "otmetka_oznakomlenija": null, "EOS_PUBLISHED_NUMBER": null, "EOS_PUBLISHED_DATE": null, "EOS_PUBLISHED_POINT": null, "EOS_PUBLISHED_CONTROL": null, "eos_item_principal": null, "eos_ir_sending": null, "MEDO_DOC_KIND": null, "bgfdbfd": null}]}]}
```

2.3 Примеры запросов

Допустим, сервис зарегистрирован по адресу <https://delo.server.org/> (далее <адрес сервиса>).

Комментарии к результатам находятся после соответствующего поля между символами \diamond и в реальный результат запроса не входят.

2.3.1 Поиск по номеру документа

Формируется следующий JSON для параметра criteries, номер документа Л-1:

```
{"Rc.Regnum":"Л-1"}
```

Запрос:

```
<адрес сервиса>/CoreHost/OData/DOC_RC?criteries=%7B"Rc.Regnum"%3A"Л-1"%7D
```

Результат:

```
{"OData.metadata":"../$metadata#DOC_RC", "$inlinecount":1, "$TotalRecords":1, "value":
[{"ISN_DOC":4363, "KIND_DOC":3, "DUE_DOCGROUP":"0.2U9.2UX.", "DOC_DATE":"2012-04-
27T00:00:00", "ORDER_NUM":0, "FREE_NUM":"Л-1", "ANNOTAT":"Об
аренде", "CONSISTS":"6", "SPECIMEN":"1", "DUE_CARD_REG":"0.", "ISN_CABINET_REG":40
89, "ISN_NOMENC":null, "SECURLEVEL":1, "ISN_DELIVERY":null, "ISREPEAT":null, "ISCOL
LECTIVE":null, "ADRESS_FLAG":1, "TEL_NUM":null, "ANONIM":null, "DUE_PERSON_EXE":"0
.2SH.2TL.", "CONTROL_STATE":null, "PLAN_DATE":null, "FACT_DATE":null, "NOTE":null,
"FILE_COUNT_VISIBLE":0, "NOTHARDCOPY":0, "CITO":0, "E_DOCUMENT":0, "ACCESS_MODE":0
, "INS_WHO":null, "INS_DATE":null, "OTHER_WHO":null, "FREE_NUM_SORT":null, "SENDER_
NUM":null, "EDIT_REASON":null, "ISN_DOCVID":0}]}
```

2.3.2 Получение объекта по идентификатору, включая вложенные списки

Получение объекта DOC_RC (регистрационной карты) по идентификатору.

Запрос:

```
<адрес
сервиса>/CoreHost/OData/DOC_RC(4709)?$expand=ACQUAINTANCE_List,AR_RC_VALUE_Lis
t
```

Результат:

```
{"OData.metadata":"../$metadata#DOC_RC/@Element",
"ISN_DOC":4709,
...
"FREE_NUM":"1-P",
...
"ACQUAINTANCE_List":[],
"AR_RC_VALUE_List":[{"ISN_RC":4709, "eos_item_number":null,... }]}
```

2.3.3 Изменение полей объекта

Изменение полей объекта производится через пакетную команду MERGE.

Запрос (метод запроса POST, ContentType: application/x-www-form-urlencoded; charset=utf-8):

```
<адрес сервиса>/CoreHost/OData/$changes
```

Пакет команд из тела запроса:

```
[{
"method": "MERGE",
"url": "DOC_RC(406057969) ",
"data": {"NOTE":"Примечание"}
}]
```

Результат:

```
[{"statusCode":204}]
```

Получение новых значений:

<адрес сервиса>/CoreHost/OData/DOC_RC(406057969)

Результат:

```
{ "OData.metadata": "../$metadata#DOC_RC/@Element",
  "ISN_DOC": 406057969,
  ...
  "NOTE": "Примечание",
  ...
}
```

2.3.4 Поиск документов по текстовым данным

Например, надо найти все документы, в которых текст отчета содержит строку 'Их наличие обеспечивало'.

Критерии запроса (параметр criteries) в формате JSON:

```
{ "Reply.Text": "\"Их наличие обеспечивало\"" }
```

Запрос (параметры URL encoded):

```
<адрес
сервиса>/CoreHost/OData/DOC_RC?criteries=%7b%22Reply.Text%22%3a%22%5c%22d%98
%d1%85+%d0%bd%d0%b0%d0%bb%d0%b8%d1%87%d0%b8%d0%b5+%d0%be%d0%b1%d0%b5%d1%81%d0%
bf%d0%b5%d1%87%d0%b8%d0%b2%d0%b0%d0%bb%d0%be%5c%22%22%7d
```

Результат:

```
{ "OData.metadata": "../$metadata#DOC_RC", "$inlinecount": 0, "$TotalRecords": 0, "va
lue":
[
{ <первый объект>
  "ACCESS_MODE": 0,
  "ADRESS_FLAG": 0,
  "ANNOTAT": "в, удивляясь тому, долину, на нежно-голубом небе ни облачка. Но
воздух холодный. О Мейлихмыкнул Уотсон. Лучше пойти, мистер Хаммел. Мы оставим
здесь несколько человек. Такстоянку",
  ...
  "SECURLEVEL": 1,
  "SPECIMEN": "1"
},
{ <второй объект>
  "ACCESS_MODE": 0,
  "ADRESS_FLAG": 0,
  "ANNOTAT": "ый код предохранительного устройства. Значит, я должен
егогранатами, чтобы",
  "ANONIM": 0,
  ...
  "PLAN_DATE": "2007-08-10T00:00:00",
  "SECURLEVEL": 1,
  "SPECIMEN": "1"
}
]
}
```

2.3.5 Добавление записей в справочник

Допустим, надо добавить новую запись в справочник регионов (REGION_CL).

Добавление, удаление и изменение данных производятся только через пакетные команды. Пакет команд помещается в тело POST запроса с Content-Type: application/x-www-form-urlencoded; charset=UTF-8 в кодировке UTF8. Адрес (всегда одинаковый для пакетных команд):

<адрес сервиса>/CoreHost/OData/\$changes

Пакет команд:

```
[ {
  "url": "REGION_CL",
  "method": "POST",
  "data": {
    "DUE": "0.-10000",
    "CLASSIF_NAME": "Край света"
  }
}]
```

Результатом выполнения пакетных команд всегда является массив в формате JSON, где каждый элемент – результат выполнения соответствующей команды. Для команды добавления записи – это назначенный новой записи идентификатор. В данном случае идентификатор строковый и содержащий в себе иерархическую структуру справочника.

Результат:

```
[
{"statusCode":201,"data":{"OData.metadata":"../$metadata#MapItemString","TempISN":null,"FixedISN":null,"ID":"0.2EZIZ.", "TempID":"0.-10000"}}]
```

Проверяем, что такой объект был добавлен. Запрос:

<адрес сервиса>/CoreHost/OData/REGION_CL(%270.2EZIZ.%27)

Результат:

```
{"OData.metadata":"../$metadata#REGION_CL/@Element","DUE":"0.2EZIZ.", "CODE":null,"COD_OKATO":null,"ISN_LCLASSIF":4058459,"PARENT_DUE":"0.", "ISN_NODE":4058459,"IS_NODE":0,"ISN_HIGH_NODE":0,"WEIGHT":4058459,"CLASSIF_NAME":"Край света","PROTECTED":0,"DELETED":0,"NOTE":null}
```

2.3.6 Получение объектов по списку идентификаторов

Для получения объектов по списку идентификаторов список указывается в параметре `ids`. Идентификаторы указываются через запятую (,), если идентификатор строковый, то он указывается в одинарных кавычках. Например (объект DEPARTMENT): '0.2SF.2T7.2TB.', '0.2SF.2T7.2TD.', '0.2SF.2T9.2TF.'

Запрос:

<адрес сервиса>/CoreHost/OData/DEPARTMENT?ids=%270.2SF.2T7.2TB.%27%2C%270.2SF.2T7.2TD.%27%2C%270.2SF.2T9.2TF.%27

Результат:

```
{"OData.metadata":"../$metadata#DEPARTMENT","$inlinecount":3,"value":[
{"DUE":"0.2SF.2T7.2TB.",
...
"CLASSIF_NAME":"Фалеев А.Д. - Нач. отдела № 1",
...},
{"DUE":"0.2SF.2T7.2TD.",
...
"CLASSIF_NAME":"Кречетов А.В. - Ведущий специалист"
...},
{"DUE":"0.2SF.2T9.2TF.",
...
"CLASSIF_NAME":"Шевченко А.Л. - Нач. отдела № 2",
...}]}
```

2.3.7 Получение списков объекта

Допустим, необходимо получить значение отдельного списка объекта, например, для оптимизации обмена данными. Имена списков внутри объекта заканчиваются на `_List`.

Правило:

<имя объекта>(<идентификатор>)/<имя списка>

Например:

DOC_RC(4328)/REF_ACCESS_CARD_List

Например, ссылка:

<адрес сервиса>/CoreHost/OData/DOC_RC(4328)/REF_ACCESS_CARD_List

Результат (объекты из списка REF_ACCESS_CARD_List):

```
{ "OData.metadata": "../$metadata#REF_ACCESS_CARD", "$inlinecount": 3, "value":
[
{ "ISN_REF_DOC": 4328, "DUE": "0.", "KIND_DOC": 1, "DATE_CARD": "2012-04-
27T12:41:35", "CONTROL_STATE": 1, "ISN_RESOLUTION": null, "HOLD": 1 },
{ "ISN_REF_DOC": 4328, "DUE": "0.2SF.", "KIND_DOC": 1, "DATE_CARD": "2012-04-
27T12:49:43", "CONTROL_STATE": null, "ISN_RESOLUTION": null, "HOLD": 0 },
{ "ISN_REF_DOC": 4328, "DUE": "0.2SF.2T7.", "KIND_DOC": 1, "DATE_CARD": "2012-04-
27T12:52:34", "CONTROL_STATE": null, "ISN_RESOLUTION": null, "HOLD": 0 }
] }
```

2.4 Объекты системы

Полный список объектов системы «ДЕЛО» находится в Приложение Г

Связи объектов системы «ДЕЛО» находятся в Приложение Д

В OData используются имена объектов и полей, которые точно соответствуют именам физических таблиц и их полям.

Есть также сущности, которые не существуют самостоятельно, а могут запрашиваться в контексте своей родительской сущности. Например, карточка подписантов, визирующих, карточка резолюций, рубрики не существуют вне контекста карточки документа DOC_RC.

Таким образом, DOC_RC это агрегат, который содержит данные из разных физических таблиц.

Эти таблицы в объекте Документ представлены свойством, состоящим из имени таблицы и окончания _List. Например, в DOC_RC кроме «родных» полей есть еще REF_SEND_List, REF_CORRESP_List и т.д.

3 C# ORM EOS (для ДЕЛО API 2009)

3.1 Модель данных, используемая в API фоновых задач

Модель данных строится на следующих понятиях:

Атрибут – обычно соответствует полю таблицы БД. Атрибут может быть одного из следующих типов:

- Строка
- Целое число
- Дата (и время)
- Дробное (десятичное) число
- Перечисление
- Коллекция сущностей
- Ссылка на сущность

Сущность – соответствует строке таблицы БД. Все сущности имеют общего предка – т.е. наследуются от базового класса EntityBase. Сущности, представляющие справочники, наследуются от класса ClassifEntity или от ClassifTreeEntity¹.

Коллекция сущностей – объединяет множество сущностей одного типа. Тип сущностей, находящихся в коллекции, может быть абстрактным. Соответствует набору строк таблицы.

Ссылка на сущность – представляет идентификатор сущности. Каждый класс сущности имеет соответствующий ему класс ссылки на эту сущность. Классы ссылок на сущности образуют иерархию наследования, аналогичную иерархии классов сущностей. В вершине иерархии наследования находится абстрактный класс EntityBaseRef.

Объект системы – представляет иерархическую структуру из сущностей с единственной сущностью в корне структуры – корневой сущностью объекта. Корневая сущность объекта может иметь атрибуты типа "Коллекция сущностей", в которых хранятся дочерние сущности объекта, которые, в свою очередь, также могут иметь свои дочерние сущности. Объект системы является единицей загрузки данных из БД.

Классы модели данных системы «ДЕЛО» определены в сборке Eos.Delo96.Api, а базовые классы в сборке Eos.Delo.Common.

Рассмотрим описанные понятия на примере некоторых сущностей системы (пропущенные части кода обозначены многоточием).

```
namespace Eos.Delo96.Api.Entities
{
    [Serializable]
    public class DOC_RC_Ref : Eos.Delo.Common.EntityBaseRef
    {
        public DOC_RC_Ref(int ISN_DOC) { ... }
        public int ISN_DOC { get { ... } }
    }
    [Serializable]
    public class DOC_RC_Entity : Eos.Delo.Common.EntityBase
    {
        ...
        public int ISN_DOC { get { ... } }
        public KIND_DOC KIND_DOC { get { ... } }
        public String DUE_DOCGROUP { get { ... } }
        public DateTime DOC_DATE { get { ... } }
        public String FREE_NUM { get { ... } }
        ...
        public List<AR_RC_VALUE_Entity> AR_RC_VALUE_List { get { ... } }
    }
}
```

¹ Класс ClassifTreeEntity унаследован от класса ClassifEntity, а класс ClassifEntity, в свою очередь, от класса EntityBase. Таким образом любая сущность имеет общего предка EntityBase.

```

    public List<DOC_WHO_Entity> DOC_WHO_List { get { ... } }
    public List<DOC_SIGN_Entity> DOC_SIGN_List { get { ... } }
    public List<FORWARD_Entity> FORWARD_List { get { ... } }
    public List<JOURNAL_Entity> JOURNAL_List { get { ... } }
    public List<REF_ACCESS_CARD_Entity> REF_ACCESS_CARD_List { get { ... } }
    ...
}
[Serializable]
public class FORWARD_Ref : Eos.Delo.Common.EntityBaseRef
{
    public FORWARD_Ref(int ISN_FORWARD) { ... }
    public int ISN_FORWARD { get { ... } }
}
[Serializable]
public class FORWARD_Entity : Eos.Delo.Common.EntityBase
{
    ...
    public int ISN_FORWARD { get { ... } }
    public int? ISN_DOC { get { ... } }
    ...
    public String DUE_ADRESAT { get { ... } }
}
[Serializable]
public class SECURITY_CL_Ref : Eos.Delo.Common.EntityBaseRef
{
    public SECURITY_CL_Ref(int SECURLEVEL) { ... }
    public int SECURLEVEL { get { ... } }
}
[Serializable]
public class SECURITY_CL_Entity : ClassifEntity
{
    ...
    public override int ISN_LCLASSIF { get { ... } }
    public override string CLASSIF_NAME { get { ... } }
    public int SECURLEVEL { get { ... } }
    public String GRIF_NAME { get { ... } }
    ...
}
}

```

Класс DOC_RC_Entity является сущностью, соответствующей таблице DOC_RC. Атрибутами этого объекта являются, например, свойства DOC_DATE и FREE_NUM – соответствующие полям таблицы DOC_RC. Сущность DOC_RC_Entity является головной сущностью агрегатного объекта "ПК документа". Объект является агрегатным, т.к. сущность имеет дочерние сущности, хранящиеся в атрибутах типа "Коллекция сущностей", таких как DOC_SIGN_List, FORWARD_List, и прочие.

Класс DOC_RC_Ref – это ссылка на сущность, в данном случае ссылка имеет единственный атрибут – ISN_DOC, что соответствует первичному ключу таблицы DOC_RC.

Класс FORWARD_Entity является дочерней сущностью "Журнал пересылки документа", входящей в состав агрегатного объекта "ПК документа". Эта сущность, соответствует таблице FORWARD.

Класс SECURITY_CL_Entity сущность, соответствующая таблице SECURITY_CL. Эта сущность является простым объектом "Гриф доступа", т.к. не имеет в своем составе списков дочерних сущностей. В отличие от сущностей, рассмотренных ранее, данная сущность наследуется от класса ClassifEntity, и имеет в своем составе атрибуты, общие для всех справочников системы, такие как ISN_LCLASSIF и CLASSIF_NAME².

² Для данного справочника, эти атрибуты будут иметь то же значение что и атрибуты SECURLEVEL и GRIF_NAME (и одноименные поля таблицы SECURITY_CL), соответственно

Полный перечень определенных в API сущностей системы и их связей приведен в Приложение Г, Приложение Д.

Схема базы данных включена в состав дистрибутива системы и расположена в каталоге \SDK\PDM, файл "Odelo.pdm". Для открытия файла можно воспользоваться программой Sybase PowerDesigner Viewer.

3.2 Соединение с базой данных

Для подсоединения поставщика данных используются методы класса Facade.

```
Facade.Open(String.Empty);
Facade.SetUserIsn(GetParam<int>("UserID"));
...
WorkContext.DropCurrent();
```

Facade.SetUserIsn устанавливает пользователя для текущего контекста соединения с БД, от имени которого будет работать ФЗ. Параметр UserID – пользователь по умолчанию, берется из настроек домена (пула) ФЗ - WAGENT. Если необходимо использовать другого пользователя, в настройках ФЗ можно назначить свой параметр ISN пользователя (название параметра любое, кроме "UserID"), и указать в параметре метода.

Для закрытия рабочего контекста используется метод WorkContext.DropCurrent().

Работа с транзакциями осуществляется с помощью интерфейса IHead. Методы интерфейса описаны в Приложение Б.

Пример:

```
IHead h = WorkContext.Current.Head;
h.BeginDbTran();
try
{
    ...
    h.Commit();
}
catch
{
    h.Rollback();
    throw;
}
```

3.3 Поиск объектов

Для поиска объектов используется метод Search класса EntityHelper. Метод возвращает список сущностей, поэтому необходимо объявить переменную, в которую будет записан результат поиска. Если сущности не найдены, возвращается пустой список. Параметром метода являются запрос типа Query<T> или критерии поиска типа SearchArguments. Первый вариант с запросом является более удобным в использовании. Поиск объектов происходит по головным сущностям системы, в результат поиска загружаются головные сущности, без загрузки дочерних, т.к. подгрузка последних приводит к проблемам с производительностью.

Поиск с Query<T>

Для экземпляра класса Query<T> указывается тип искомого объекта. Критерии поиска задаются с использованием методов SetCriterion и TextCriterion.

Пример поиска объекта. Следующие три критерия задают поиск резолюции, которая имеет признак контрольной, относится к определённой РК, и дата которой находится в заданном диапазоне:

```
var q = new Query<RESOLUTION_Entity>();
q.SetCriterion(e => e.ISN_REF_DOC, RcEntity.ISN_DOC);
q.SetCriterion(e => e.CONTROL_STATE, 1);
q.SetCriterion(e => e.RESOLUTION_DATE, "01/01/2020:10/12/2021");
var result = q.Search();
```

Если предполагается загрузка большого количества данных, необходимо ограничивать количество записей для загрузки. Данные параметры указываются в поле SA

запроса. Пример для загрузки 200 событий, со значением ISL_EVENT от value и больше, отсортированных по полю ISL_EVENT:

```
var q = new Query<EVNT_FEED_Entity>();
q.SetCriterion(ef => ef.ISL_EVENT, string.Format("{0}:null", value));
q.SA.OrderBy = "ISL_EVENT";
q.SA.StartRowIndex = 0;
q.SA.RecordsPerPage = 200;
var result = q.Search();
```

Вложенный поиск. Зачастую недостаточно указать критерии поиска со свойствами, соответствующими полям таблицы БД сущности, может понадобиться поиск по свойствам дочерних сущностей. Критерии вложенного поиска указываются в поле SA запроса. Рассмотрим пример поиска по доп. реквизитам для РК документов:

```
var q = new Query<DOC_RC_Entity>();
q.SetCriterion(e => e.DUE_DOCGROUP, "0.2TZ.2EZDG.");
q.SetCriterion(e => e.DOC_DATE, "01/05/2021:07/07/2021");
q.SA.Criteria["AR_RC_VALUE.end_dog"] = "1";
q.SA.Criteria["AR_RC_VALUE.sum_dog"] = "250000:350000";
q.SA.Criteria["AR_RC_VALUE.date_end"] = "08/07/2021";
q.SA.Criteria["ACQUAINTANCE.DUE_ADRESAT"] = "0.2SD.2SP.";
var docRcList = q.Search();
```

В данном примере поиск РК осуществляется по группе РК с идентификатором "0.2TZ.2EZDG. " и периоду создания РК с 1 мая 2021 года по 7 июля 2021 года. Вложенный поиск по доп. реквизитам (AR_RC_VALUE) содержит три критерия, end_dog – флаг закрытого договора, где 1 означает, что флаг взведен, sum_dog – сумма договора, от 250 000 до 350 000 и дата завершения договора 8 июля 2021 года. Вложенный поиск по журналу ознакомления содержит один критерий DUE_ADRESAT – идентификатор адресата, равный "0.2SD.2SP.". После выполнения поиска, поле docRcList содержит список сущностей РК документа типа DOC_RC_Entity или пустой список.

Поиск протоколов. Поиск протоколов производится по сущности FULL_PROT_Entity. Пример поиска записей протокола по изменениям определенной РК, отсортированный по времени, полю TIME_STAMP:

```
var q = new Query<FULL_PROT_Entity>(LoadMode.Table);
q.SA.OrderBy = "TIME_STAMP";
q.SetCriterion(p => p.TABLE_ID, "D");
q.SetCriterion(p => p.OPER_ID, "U");
q.SetCriterion(p => p.SUBOPER_ID, "U");
q.SetCriterion(p => p.REF_ISN, RcEntity.ISN_DOC);
var protocols = q.Search();
```

Поиск с SearchArguments

Критерии поиска задаются в виде экземпляра класса SearchArguments. Обязательно должно быть заполнено свойство Result данного класса, задающее объект, который нужно искать. Значение, задаваемое в этом свойстве должно быть строкой, соответствующей имени искомой сущности без суффикса "_Entity". После задания типа результата поиска, нужно задать критерии поиска.

Пример поиска объекта, аналогичный поиску по Query:

```
var sa = new SearchArguments();
sa.Result = "RESOLUTION";
sa.Criteria.Add("RESOLUTION.RESOLUTION.ISN_REF_DOC",
RcEntity.ISN_DOC.ToString());
sa.Criteria.Add("RESOLUTION.RESOLUTION.CONTROL_STATE", "1");
sa.Criteria.Add("RESOLUTION.RESOLUTION.RESOLUTION_DATE",
"01/01/2020:10/10/2021");
var resolutionListSA = EntityHelper.Search<RESOLUTION_Entity>(sa);
```

*** Примечание.** Рекомендуется использовать тип var при инициализации объектов, в связи с ожидаемой сменой типов идентификаторов.

3.4 Поиск ссылки на сущность системы

Когда необходимо найти ссылку на сущность системы, используется метод SearchRef класса EntityHelper. Для метода указывается тип искомого объекта – ссылка на сущность, оканчивается как “_Ref”. Параметром метода являются критерии поиска, которые задаются в виде экземпляра класса SearchArguments, как и для поиска объектов.

Пример поиска ссылки на вид доставки по его названию. Ссылка на вид доставки содержит единственное поле ISN_LCLASSIF, соответствующее полю первичного ключа ISN_LCLASSIF в базе данных для данной сущности.

```
var q = new Query<DELIVERY_CL_Entity>();
q.TextCriterion(d => d.CLASSIF_NAME, "Почта", true);
var deliveryIsn = EntityHelper.SearchRef<DELIVERY_CL_Ref>(q.SA).Select(d =>
d.ISN_LCLASSIF).FirstOrDefault();
```

В данном примере в параметр метода передается экземпляр класса SearchArguments, являющийся полем класса Query<T>.

3.5 Загрузка объекта по идентификатору

Часто при разработке фоновых задач необходимо получить (загрузить) данные единственного экземпляра объекта по известному системному коду. Для загрузки используется метод Load класса EntityHelper. Метод имеет несколько перегрузок: поиск объекта по его ссылке, поиск нескольких объектов по списку ссылок, а также оба варианта с указанием опции загрузки – элементом класса LoadOptions.

Для получения (загрузки) объекта по известному коду, необходимо создать из кода ссылку на сущность – объект одного из типов _Ref. В большинстве случаев, нужная ссылка доступна как свойство уже загруженной ранее сущности. Например, если загружена РК документа, и требуется проанализировать ее группу документов, нужно объявить переменную для хранения сущности DOCGROUP_CL_Entity, в которую будет помещён результат загрузки.

Пример загрузки объекта РК документа по известному числовому значению кода и группы документов по ссылке, содержащейся в объекте РК документа:

```
var rcEntity = EntityHelper.Load<DOC_RC_Entity>(new DOC_RC_Ref(4954));
var docgroup = EntityHelper.Load<DOCGROUP_CL_Entity>(rcEntity.DOCGROUP_Ref);
```

3.6 Модификация данных объекта

Зачастую, фоновые задачи создаются для выполнения различных модификаций объектов системы. Для модификаций объекта реализованы специальные классы-помощники для объектов, это классы, содержащие имя объекта и оканчивающиеся на “_Helper”, так для РК это DOC_RC_Helper, для резолюции RESOLUTION_Helper и т.д.

У каждого такого “_Helper” класса реализованы свои методы для модификации объекта. Например, для РК, в классе DOC_RC_Helper есть методы ApplyDeliveryRule – для заполнения вида доставки, Append_REF_FILE – для прикрепления документа, Append_REF_LINK – добавление связки и др. Для резолюции - Append_REPLY- для добавления исполнителя и др.

При создании экземпляра класса “_Helper” указываются свойства Changes – экземпляр класса Changes и объект для модификации, причем объект должен быть загружен с возможностью редактирования, методом Load с указанием опции LoadOptions.ForEdit или методом LoadForEdit класса EntityHelper.

Методы Changes.Put(EntityBase entity) - регистрирует головную сущность объекта в наборе изменений, Changes.Save() - сохраняет данные Changes.

Пример добавления рубрики к РК документа:

```
var rcEntity = EntityHelper.Load<DOC_RC_Entity>(new DOC_RC_Ref(4954),
LoadOptions.ForEdit);
var helper = new DOC_RC_Helper()
{
    DocRc = rcEntity,
```

```

    Changes = new Changes()
};
helper.Changes.Put(helper.DocRc);
helper.Append_REF_RUBRIC(new RUBRIC_CL_Ref(dueRubric));
helper.Changes.Save();

```

Для более простых модификаций возможно использовать сохранение измененного объекта с помощью экземпляра класса `Changes`, без использования класса-помощника. Например, если необходимо поменять один из атрибутов сущности.

Пример изменения признака «на контроле» резолюции:

```

var resolution = EntityHelper.LoadForEdit<RESOLUTION_Entity>(resolutionRef);
resolution.CONTROL_STATE = 1;
var changes = new Changes();
changes.Put(resolution);
changes.Save();

```

Некоторые классы-помощники, например, для РК документов, РКПД, резолюций (поручений) содержат метод `Create`, который регистрирует новый объект. Для РК и РКПД указывается группа документов, для которой создается новый объект, для резолюции вид резолюции (поручение или пункт) и ссылка на родительскую резолюцию.

Пример создания РК с резолюцией:

```

private void DocRcCreate(string dueDocGroup, string executorDue, string
authorDue, string filePath)
{
    var DateTimeNow = DateTime.Now;
    var docRcHelper = new DOC_RC_Helper()
    {
        ViewContext = DOC_RC_SaveContext.Current,
        Changes = new Changes(),
        User = UserContext.Current.USER_CL
    };

    var docRC = docRcHelper.Create(new DOCGROUP_CL_Ref(dueDocGroup));
    docRC.DOC_DATE = DateTime.Today;
    docRC.FREE_NUM = Guid.NewGuid().ToString("N");
    docRC.AR_RC_VALUE_List[0].SetVal("sum_dog", 350000);
    docRC.CONSISTS = "1";
    docRC.ANNOTAT = "Содержание РК";

    docRcHelper.ApplyDefaultRules();
    AttachFile(docRcHelper, filePath);

    var resHelper = new RESOLUTION_Helper(docRcHelper);
    var resolution = resHelper.Create(1, null);

    resHelper.ApplyDefaultRules();
    resHelper.Append_REPLY_Department(new DEPARTMENT_Ref(executorDue));

    resolution.DUE = authorDue;
    resolution.RESOLUTION_DATE = DateTimeNow;
    resolution.SEND_DATE = DateTimeNow;
    resolution.LEFT_RESOLUTION = 0;
    resolution.RESOLUTION_TEXT = "Текст резолюции";
    docRcHelper.Changes.Save();
}

private void AttachFile(DOC_RC_Helper docRcHelper, string tempFilePath)
{
    string tempFileName = Path.GetFileName(tempFilePath);
    string fileName = Path.GetFileNameWithoutExtension(tempFileName);

    var refFile = docRcHelper.Append_REF_FILE(fileName, "1");
    refFile.DESCRPTION = tempFileName;
    refFile.NAME = tempFileName;
}

```

```
refFile.SetContents(tempFilePath);
}
```

*** Примечание.** Рекомендуется запоминать сущности в коде вместо многократного прохода через точки. Например, присваивание состава РК как `docRC.CONSISTS = "1"`; где `var docRC = docRcHelper.DocRc`, вместо `docRcHelper.DocRc.CONSISTS = "1"`;

В данном примере создает РК с доп. реквизитом “sum_dog”, составом (CONSISTS), содержанием (ANNOTAT), добавлением правил по умолчанию (ApplyDefaultRules), файла и резолюции. Для резолюции указаны автор (DUE), дата резолюции (RESOLUTION_DATE), дата рассылки (SEND_DATE), признак рассылки в папку "На контроле" – контролеру и автору, в "Поступившие" и "На исполнении" – исполнителям (LEFT_REZOLUTION = 0) и текст резолюции (RESOLUTION_TEXT).

3.7 Выполнение хранимых процедур БД

Для выполнения хранимой процедуры используется метод Execute интерфейса IHead. При этом пользователь от имени которого открывается связь с БД должен иметь права на выполнение процедуры, иначе создается исключение.

Соединение с БД обязательно должно выполняться с заданием текущего пользователя. Все изменения в БД вносятся от имени заданного пользователя.

4 GraphQL (для ДЕЛО API 2023)

4.1 Описание протокола

В качестве протокола обмена информацией используется протокол HTTP.

Используемые методы протокола HTTP: GET, POST.

Обработку запросов осуществляет приложение на web-сервере, предоставляющее доступ к API и работающий на платформе EOS Platform Host.

Допустим, приложение зарегистрировано по адресу <https://delo.server.org/> (далее <адрес приложения>).

Обращения к API осуществляются через <адрес приложения>/CoreHost/<адрес контроллера>/<путь и параметры запроса>.

Для обработки запросов REST API, включая загрузку и скачивание файлов, доступен один контроллер по адресу:

- gql/query - для обработки запросов чтения и манипулирования объектами системы через GraphQL,

Все запросы на контроллер gql/query, как на чтение, так и на изменение данных идут через метод POST.

Запрос на загрузку файла gql/query - POST, запрос на чтение файла gql/query/download - GET.

Результат запроса (если не определено иное) будет представлен в виде объекта или массива объектов в формате JSON в теле ответа с **Content-Type:application/json; charset=UTF-8**.

Примеры формирования запросов написаны с использованием PowerShell

Запросы могут быть с параметрами и без параметров.

Создание сессии и закрытие сессии в системе «ДЕЛО» являются универсальными для ДЕЛО API 2009 (OData) и ДЕЛО API 2023 (GraphQL).

Описание и примеры кода см. 8.1. API входа в систему «ДЕЛО» и выхода из системы.

4.1.1 Пример выполнения запроса без параметров

```
# Выполнение запроса без параметров
$text = '{
  getDeliveryCl(isnLclassif:-10000)
  {
    classifName
    protected
  }
}'

$json = @{query = $text; variables = $null; } | ConvertTo-Json

$Response = Invoke-WebRequest -Uri "$BaseUrl/CoreHost/gql/query" -Body $json -
Method 'POST' -WebSession $Session -ContentType "application/json;
charset=utf-8"
```

4.1.2 Пример выполнения запроса с параметрами

```
# Выполнение запроса с параметрами
$text = 'mutation SaveBlob($fileId1: String!, $filename1: String!) {
  createl: createRefFile (input:
  {
    clientMutationId:"sid",
    data: {
      isnObject: 4377,
      refFileTypeNames: "RefFilePrj",
      contents: $fileId1,
      description: $filename1
    }
  }
}
```

```

    }
  }
)
{
  success
  message
  messageCode
}
}'

$vars = '{ "fileId1": "$fileId1", "filename1": "$filename1" }'

$operName = 'Save'

$json = @{'query' = $text; 'variables' = $null; 'operationName' = $operName; } |
ConvertTo-Json

$Response = Invoke-WebRequest -Uri "$BaseUrl/CoreHost/gql/query" -Body $json -
Method 'POST' -WebSession $Session -ContentType "application/json;
charset=utf-8"

```

Далее в примерах будем рассматривать только формирование текста запроса без параметров, а не полный json запросов.

4.1.3 Утилита с графическим интерфейсом для изучения GraphQL (схемы и отладки запросов)

В систему «ДЕЛО» встроена утилита с графическим интерфейсом, с помощью которой можно изучать схему данных системы, доступные запросы и мутации, а также писать и отлаживать запросы на GraphQL.

Чтобы воспользоваться этой утилитой нужно:

- зайти в систему через стандартную страницу Вход в систему: **<адрес приложения>/login/**
- в адресной строке браузера ввести путь к утилите: **<адрес приложения>/CoreHost/graphiql**

4.1.4 Правилами формирования имен сущностей, свойств, методов, запросов.

Правила формирования имен сущностей и свойств из имен таблиц и колонок БД соответствуют стилю написания составных слов camelCasingStyle.

Имена сущностей и полей образуются из имен таблиц и колонок по следующим правилам:

- каждая часть имени таблицы (колонки), отделенная символом подчеркивания, записывается с большой буквы;
- символы подчеркивания исключаются;
- первая часть имени пишется с маленькой буквы.

Для каждой сущности системы «ДЕЛО» в GraphQL существует метод, для получения отдельного экземпляра сущности по ее первичному ключу.

Имя метода образуется по шаблону 'get'+<Имя сущности>, параметры метода соответствуют свойствам первичного ключа. Например: getDeliveryCl.

Также для каждой сущности системы «ДЕЛО» в GraphQL существует метод, для получения частичного списка сущностей определенного типа.

Имя метода совпадает с именем сущности в множественном числе с добавлением окончания Pg для страничной выборки, и Cr для курсорной выборки. (пейджинга). Например: refFilesPg, refFilesCr.

Для запросов на изменение данных (добавление, изменение, удаление) имя запроса формируется по следующим шаблонам:

- добавление: 'create'+<Имя сущности> (например, createDeliveryCl);

- изменение: 'update'+<Имя сущности> (например, updateDeliveryCl);
 - удаление: 'delete'+<Имя сущности> (например, deleteDeliveryCl).
- Полный список объектов системы «ДЕЛО» находится в Приложение Г
Связи объектов системы «ДЕЛО» находятся в Приложение Д.

4.2 Запросы GraphQL на получение данных (query)

Первый вид запросов к GraphQL - query, подобные запросы предназначены для получения данных.

Запросы query могут быть на получения всех данных таблицы, на получение одной записи, запросы на страничный и курсорный пейджинг.

4.2.1 Пример запроса на получение одной записи:

```
# Запрос на получение одной записи
$text = '{
  getDeliveryCl(isnLclassif:-10000)
  {
    classifName
    protected
  }
}'
```

4.2.2 Запросы на страничный и курсорный пейджинг

Запросы страничного и курсорного пейджинга позволяют формировать сложные условия выборки с несколькими условиями, операторами EXISTS, сортировкой и т.д.

Пример запроса на страничный пейджинг:

```
# Запрос на страничный пейджинг
$text = '{
  refFilesPg(refFileTypeNm: "RefFileTempRc" filter:{and: [ {
isnObject:{equal:{value:' + $isnTempRc + '}}},
  { kindDoc:{equal:{value:701}} } ] })
  {
    items
    {
      isnRefFile
    }
  }
}'
```

Пример запроса на курсорный пейджинг:

```
# Запрос на курсорный пейджинг
$text = '{
  refFilesCr(refFileTypeNm: "RefFileTempRc" filter:{and: [ {
isnObject:{equal:{value:' + $isnTempRc + '}}},
  { kindDoc:{equal:{value:701}} } ] })
  {
    items
    {
      isnRefFile
    }
  }
}'
```

4.2.3 Запросы с вложенными объектами

Можно делать сложные запросы с вложенными объектами:

```
# Сложный запрос с вложенными объектами
$text = '{
  getDepartment(due:"0.2SH.2TJ.")
  {
    due
    parentNode {
```



```

    due
    parentNode {
      due
    }
  }
}
}'

```

Результатом будет:

```

{
  "data": {
    "getDepartment": {
      "due": "0.2SH.2TJ.",
      "parentNode": {
        "due": "0.2SH.",
        "parentNode": {
          "due": "0."
        }
      }
    }
  }
}

```

4.2.4 Представление результата

Результат запроса (если не определено иное) будет представлен в виде объекта в формате JSON в теле ответа с **Content-Type:application/json; charset=UTF-8**.

Результаты запросов на чтение представлены совокупностью скалярных полей (строка, число и т.д.), а также вложенных объектов и списков объектов.

Вложенные объекты это сущности, связанные с сущностью первого уровня отношением многие-к-одному.

Для этих объектов имена методов, как правило, совпадают с именами этих сущностей.

В иерархических справочниках для связи с вершиной верхнего уровня существует метод parentNode.

Пример:

```

{
  "data": {
    "getDepartment": {
      "due": "0.2SH.",
      "isNode": 0,
      "isnNode": 3616,
      "note": null,
      "parentNode": {
        "due": "0.",
        "isnNode": 0,
        "isNode": 0,
        "note": null
      },
      "childNodes": [
        {
          "due": "0.2SH.2TJ.",
          "isnNode": 3654,
          "isNode": 1,
          "note": null
        },
        {
          "due": "0.2SH.2TL.",
          "isnNode": 3656,
          "isNode": 1,
          "note": null
        }
      ]
    }
  }
}

```

```

    ]
  }
}

```

Запросы GraphQL завершаются с кодом ошибки HTTP 400 или 500 только в том случае, если запрос не верно сформирован, не открыта сессия пользователя, или другие подобные проблемы, не позволяющие службе приступить к выполнению запроса.

Во всех остальных случаях код возврата HTTP будет 200, даже если в процессе выполнения запроса возникли ошибки. Информацию об ошибках, возникших в процессе выполнения запроса нужно получать из объекта результата выполнения в поле errors.

4.3 Запросы GraphQL на изменение данных (Mutation)

Запросы на изменение данных позволяют выполнять добавление, изменение и удаление данных.

Важным параметром запроса на изменение данных является clientMutationId, который позволяет "связать" запрос, отправленный на сервер с ответом сервера на клиенте.

4.3.1 Добавление данных

Пример добавления:

```

# Запрос на изменение данных - добавление
$text = 'mutation
{
  createDeliveryCl(input:
  {
    clientMutationId:"d"
    data:{
      classifName: "тест"
      protected:0
      deleted:0
    }
  }) {
    success
    message
    messageCode
    systemMessage
    data {
      isnLclassif
    }
  }
}'

```

В случае возникновения ошибки будут возвращены ошибка, код ошибки, системная ошибка в соответствующих полях.

4.3.2 Изменение данных

Пример изменения:

```

# Запрос на изменение данных
$text = 'mutation
{
  updateDeliveryCl(input:
  {
    clientMutationId:"d"
    data:{
      isnLclassif: -10000
      classifName: "тест123"
      note: "Текст комментария"
    }
  }) {
    success
    message
  }
}'

```

```

        messageCode
        systemMessage
    data {
        isnLclassif
    }
}
}'

```

4.3.3 Удаление данных

Пример удаления:

Запрос на изменение данных - удаление

```

$text = 'mutation {
  deleteDeliveryCl(input:
    {
      clientMutationId:"d"
      pk:{
        isnLclassif: -10000
      }
    }) {
    success
    message
  }
}'

```

Пример удаления нескольких сущностей:

Запрос на изменение данных - удаление нескольких сущностей

```

$text = 'mutation deleteFiles($fileId1: Long!, $fileId2: Long!) {
  delete: deleteRefFile(input:
    {
      clientMutationId: "id"
      pks: [{ isnRefFile: $fileId1 }, { isnRefFile: $fileId2 }]
    })
    {
      success
      message
      messageCode
    }
}'

```

4.3.4 Вложенные объекты

Пример сложного запроса (создание РКПД) на добавление с созданием родительских объектов:

Сложный запрос (создание РКПД) на добавление с созданием родительских объектов

```

$text = 'mutation {
  createPrjRc(input:
    {
      clientMutationId:"prjrc"
      data:{
        dueDocgroup: "" + $due + ""
        securlevel: 1
        prjDate: "2022.05.01"
        annotat: "тест"
        note: "тест"
        isnNomenkl: 4057157
        prjExecs: [
          {
            createPrjExec: {
              addInfo: "new exec",
              duePerson: "" + $DueDep + "",
              canManageApproval: 1,
              canManageEndorse: 1,
              canManageExec: 1,

```

```

        canManageProcess: 1,
        canManageSign: 1,
        canWorkWithFiles: 1,
        canWorkWithPrj: 1,
        isAuthor: 1
    }
}
}
}))
{
    success
    message
    messageCode
    systemMessage
    data {
        isnPrj
    }
}
}

```

4.3.5 Представление результата

Результатом мутации являются следующие поля (см. Таблица 1):

Таблица 1

Наименование	Описание
clientMutationId	Связывающий идентификатор
success	Признак успешного выполнения
message	Сообщение об ошибке (в случае успешного выполнения - null)
messageCode	Код бизнес-ошибки (в случае успешного выполнения - null)
systemMessage	Системное сообщение об ошибке (в случае успешного выполнения - null)
data { ... }	Возвращает необходимые клиенту столбцы, например идентификатор созданного объекта и т.д.

4.3.6 Кастомные аргументы

Кастомные аргументы позволяют передавать в мутацию дополнительную информацию, которая затем используется в конвейере middleware (например, для дополнительных проверок или дополнительных действий)

Пример:

```

# Передача в мутацию кастомных аргументов с дополнительной информацией
// Регистрация документа на основе проекта
$text = 'mutation {
  createDocRc(input:
    {
      clientMutationId:"docrc"
      data:{
        dueDocgroup: "0.2U9.2UX."
        docDate: "2021.09.01"
        ...
      }
    }
  registerDocFromPrj: {
    isnPrj: 123
  })
}'

```

```
{
  success
  message
  messageCode
}
```

В приведенном выше примере в мутацию передается пользовательский аргумент registerDocFromPrj (регистрировать документ на основе РКПД).

Список кастомных аргументов можно найти в 4.7.2. Кастомные аргументы.

4.3.7 Пакетные команды

GraphQL позволяет создавать пакетные команды, с помощью которых можно добавить сразу несколько сущностей в рамках одного пакета.

Пример:

```
# Пакетные команды - в примере создаются 2 файла в рамках одного пакета
$text = 'mutation SaveBlob($fileId1: String!, $fileId2: String!, $filename1:
String!, $filename2: String!) {
  create1: createRefFile (input:
    {
      clientMutationId:"sid1",
      data: {
        isnObject: 4377,
        refFileTypeNm: "RefFilePrj",
        contents: $fileId1,
        description: $filename1
      }
    }
  )
  {
    success
    message
    messageCode
  }

  create2: createRefFile (input:
    {
      clientMutationId:"sid2",
      data: {
        isnObject: 4377,
        refFileTypeNm: "RefFilePrj",
        contents: $fileId2,
        description: $filename2
      }
    }
  )
  {
    success
    message
    messageCode
  }
}
```

В приведенном выше примере создаются 2 файла в рамках одного пакета.

4.4 Работа с файлами: скачивание файлов

Для получения файла сперва необходимо получить специальный объект BlobHandle: getRefFile(isnRefFile: \$id refFileTypeNm: "...") { contents }. Затем необходимо вызвать GET метод контроллера и передать туда handle - gql/query/download?handle=\$contents

```
# чтение файла
$text = '
```

```

    {
        getRefFile(isnRefFile: ' + $IsnRefFile + ' refFileType:
"RefFilePrj") {
            contents
        }
    }
    $json = @{query = $text; variables = $null;} | ConvertTo-Json
    $Response = Invoke-WebRequest -Uri "$BaseUrl/CoreHost/gql/query" -Body
    $json -Method 'POST' -WebSession $Session -ContentType "application/json;
    charset=utf-8"

    $content = ($Response.Content | ConvertFrom-Json).data.getRefFile.contents

    $Response = Invoke-WebRequest -Uri
    "$BaseUrl/CoreHost/gql/query?handle=$content" -Method 'GET' -WebSession
    $Session

```

4.5 Примеры запросов

4.5.1 Создание Contact и Department

Создание Contact и Department

```

$text = 'mutation
{
    createContact (
        input: {
            clientMutationId: "cdp",
            data: {
                isnContact: 5480
                isnOrganiz: 0
                deleted: 0
                edsFlag: 0
                encryptFlag: 0
                techDepartment: {
                    createDepartment: {
                        classifName: "тест"
                        isNode: 1
                        isnOrganiz: 0
                        isnCabinet: null
                        protected: 0
                        deleted: 0
                        expeditionFlag: 0
                        numcreationFlag: 0
                    }
                }
            }
        }
    ) {
        success
        message
        messageCode
    }
}'

```

4.5.2 Запрос протокола через курсорный пейджинг

Запрос протокола через курсорный пейджинг

```

$text = 'query TestQuery {
    prots: protsCr(filter: { tableId: { equal: { value: "P" } } }) {
        items {
            operDescribe
            refIsn
            suboperId
            userIsn
        }
    }
}'

```

```
}
}'
```

4.5.3 Создание РКПД

```
# Создание РКПД
$text = 'mutation {
  createPrjRc(input:
    {
      clientMutationId: "prjrc"
      data: {
        dueDocgroup: "0.2UB.2UD.2UF."
        securlevel: 1
        prjDate: "2021.09.01"
        prjExecs: [
          {
            createPrjExec: {
              addInfo: ""new exec"",
              duePerson: ""0.2SD.2SP."",
              canManageApproval: 1,
              canManageEndorse: 1,
              canManageExec: 1,
              canManageProcess: 1,
              canManageSign: 1,
              canWorkWithFiles: 1,
              canWorkWithPrj: 1,
              isAuthor: 1
            }
          }
        ]
      }
    }
  }
  prjRefRubrics: [
    {
      createPrjRefRubric: {
        dueRubric: ""0.2EYD3.2EYD5.2EYDH.""
      }
    }
  ]
}
}'
```

4.5.4 Получение подразделения и связанный SevAssociation

```
# Получение подразделения и связанный SevAssociation
$text = '{
  getDepartment(due: "0.2SD.2SP.") {
    due
    sevAssociation {
      globalId
    }
  }
}'
```

4.6 Тесты REST API GraphQL

4.6.1 Требования

Для запуска тестов необходимы следующие компоненты:

- Установленное и настроенное приложение ДЕЛО-WEB. Приложение должно быть доступно с компьютера, на котором запускаются тесты.

- Установленный Powershell версии 7.2 или старше

4.6.2 Состав пакета

Скрипты с тестами входят в состав дистрибутива системы (комплект SDK), располагаются в подкаталоге ...Sdk\Samples и включают в себя:

- Набор файлов скриптов на языке Powershell (расширение .ps1)
- Пример текстового файла для прикрепления к РК

4.6.3 Запуск тестов

Запуск тестов осуществляется из командной строки в консоли Powershell соответствующей требованиям версии (не обязательно запускать с правами Администратора, консоль можно запускать под текущей учетной записью пользователя), текущей папкой должна быть папка с файлами для тестов. При желании возможно запускать скрипты из среды отладки, например, VisualStudioCode с установленным плагином Powershell позволяет пошагово выполнять скрипты с просмотром переменных.

В начале каждого скрипта содержится блок входящих параметров со значениями по умолчанию. Можно отредактировать в текстовом редакторе значения параметров прямо в тексте скрипта либо указать их при запуске в командной строке ".\<имя скрипта>.ps1 -<имя параметра1> <значение параметра1> -<имя параметра2> <значение параметра2> ..."

4.6.4 Пример входа в систему

Данный тест демонстрирует простой вход в систему «ДЕЛО» и показ данных текущего пользователя. Следует указать параметры:

- BaseUrl – адрес сервиса
- UserName – логин пользователя
- Pass – пароль пользователя

Скрипт посылает запрос на вход в систему и выводит содержимое ответа сервера. Если вход в систему состоялся, выводится сообщение «Вход произведён», после чего осуществляется запрос данных о пользователе по Id, который вернулся в ответ на запрос о входе. Выводятся данные из ответа сервера в виде возвращаемого текста в формате Json. После посылается запрос на выход из системы, сервер в нормальной ситуации в теле ответа ничего не сообщает.

Пример:

```
PS D:\restapi> .\login.ps1
Попытка входа в систему
{
  "user": "tver",
  "password": "tver"
}
Response status code does not indicate success: 400 (Bad Request).
Ответ сервера:
{"NeedRedirect":true,"Url":"/Dev.Delo96.x2021.Site/Main.aspx","Code":null,"Message":null}
Вход произведён
{"data":{"getCurrentUserCl":{"classifName":"TVER"}}}
Выход из системы
Ответ сервера: {"NeedRedirect":false,"Url":null,"Code":null,"Message":null}
```

4.6.5 Примеры операций с элементом справочника

4.6.5.1 Добавление и изменение элемента справочника

Данный тест демонстрирует использование пакетных команд для добавления и изменения элемента справочника «Виды доставки».

При запуске следует указать параметры:

- BaseUrl – адрес сервиса

При запуске следует указать параметры:

- BaseUrl – адрес сервиса
- UserName – логин пользователя
- Pass – пароль пользователя
- DeliveryName – имя удаляемого элемента

Для того, чтобы изменение справочника прошло, у пользователя должны быть соответствующие права («Системный технолог» и право на данный справочник). В демо-примере это право есть у пользователя TVER.

Скрипт делает вход в систему. Ищет элемент с указанным именем DeliveryName. Получает Id элемента и удаляет его. Затем проводится контроль на корректность удаления, и, в зависимости от результата операции, выдается сообщение: "Элемент удалился" / "Элемент не удалился".

Пример:

```
PS D:\restapi> .\delivery_find_remove.ps1
Попытка входа в систему
{
  "user": "tver",
  "password": "tver"
}
Response status code does not indicate success: 400 (Bad Request).
Ответ сервера:
{"NeedRedirect":true,"Url":"/Dev.Delo96.x2021.Site/Main.aspx","Code":null,"Message":null}
Вход произведён
Ищем вид доставки
Критерий поиска: {
  "variables": null,
  "query": "{\r\n
deliveryClsPg(filter:{classifName:{equal:{value:\"тест123\"}}})\r\n      {\r\n
items\r\n      {\r\n      isnLclassif\r\n      }\r\n      }\r\n      }"
}
Ответ сервера: {"data":{"deliveryClsPg":{"items":[{"isnLclassif":-10000}]}}
Удаляем элемент видов доставки
Тело запроса {
  "variables": null,
  "query": "mutation {\r\n      deleteDeliveryCl(input:\r\n      {\r\n
clientMutationId:\"d\"\r\n      pk:{\r\n      isnLclassif:-10000\r\n
}\r\n      }) {\r\n      success\r\n      message\r\n      }\r\n      }"
}
Ответ сервера: {"data":{"deleteDeliveryCl":{"success":true,"message":null}}}
Ищем вид доставки
Критерий поиска: {
  "variables": null,
  "query": "\r\n      {\r\n      getDeliveryCl(isnLclassif:-10000)\r\n
{\r\n      classifName\r\n      }\r\n      }"
}
Ответ сервера: {"data":{"getDeliveryCl":null}}
Элемент удалился
Выход из системы
Ответ сервера: {"NeedRedirect":false,"Url":null,"Code":null,"Message":null}
```

4.6.6 Примеры добавления РКПД

Данный тест демонстрирует использование команд для добавления РКПД, а также загрузку файла.

При запуске следует указать параметры:

- BaseUrl – адрес сервиса
- UserName – логин пользователя
- Pass – пароль пользователя

- DGName – имя группы документов создаваемой РКПД
- FileName – имя прикрепляемого файла, должен находиться в текущей папке
- FilePath - полный путь к прикрепляемому файлу
- FileId - произвольный уникальный идентификатор файла

Для того, чтобы создание РКПД с файлом прошло, у пользователя должны быть соответствующие права. Например, необходимо право «Создание РКПД». Также у пользователя должно быть ассоциированное ДЛ, так как оно берётся в качестве исполнителя РКПД.

Также, как в 4.6.4. Пример входа в систему происходит попытка входа в систему и считываются данные о текущем пользователе, чтобы узнать ассоциированное ДЛ. После этого по имени ищется группа документов.

Пример:

```
.\add_prj.ps1 -BaseUrl <адрес сервера> -UserName kav -Pass kav -FilePath
"C:\Users\yudin\m\Desktop\корректировка БД.docx"
Попытка входа в систему
{
  "user": "kav",
  "password": "kav"
}
Получаем данные о текущем пользователе
Ответ сервера: {
  "data": {
    "getCurrentUserCl": {
      "classifName": "KAV",
      "dueDep": "0.2SF.2T7.2TD."
    }
  }
}
ID ассоциированного ДЛ: 0.2SF.2T7.2TD.
Ищем группу документов
Критерий поиска: {
  "query":
"{\r\n      docgroupClsPg(filter:{classifName:{equal:{value:\"Распоряжения\"}}
})\r\n      {\r\n          items\r\n          {\r\n              due\r\n              isnN
ode\r\n              isNode\r\n          }\r\n      }\r\n      }",
  "variables": null
}
}
Ответ сервера: {
  "data": {
    "docgroupClsPg": {
      "items": [
        {
          "due": "0.2UB.2UD.2UJ.",
          "isnNode": 3690,
          "isNode": 1
        }
      ]
    }
  }
}
Загрузка файла
Ответ сервера: 9a4af5cfed8f47bf985e4b5051fa6762
Создание РКПД
Ответ сервера: {
  "data": {
    "createPrjRc": {
      "success": true,
      "message": null,
      "messageCode": null,
      "systemMessage": null,

```

```

    "data": {
      "isnPrj": 4532
    }
  }
}
}
}
Получаем данные о сохранённой РКПД
Ответ сервера: {
  "data": {
    "getPrjRc": {
      "dueDocgroup": "0.2UB.2UD.2UJ.",
      "note": "тест",
      "prjDate": "2022-05-01T00:00:00Z"
    }
  }
}
}
Выход из системы
Ответ сервера: {
  "NeedRedirect": false,
  "Url": null,
  "Code": null,
  "Message": null
}

```

4.7 Дополнительная информация по пользовательским запросам

4.7.1 Пользовательские запросы GraphQL

В API присутствует ряд пользовательских запросов GraphQL. Ниже приведен список этих запросов.

Запросы (см. Таблица 2):

Таблица 2

Наименование	Описание
getCurrentUserCl	Получение текущего пользователя
IWapiSession.acl	Список прав доступа к сессии.
buildPrjRc	Возвращает данные новой РКПД.
buildPrjRcVersion	Возвращает данные РКПД для новой версии.
buildPrjRcRestore	Возвращает данные РКПД для восстановления из версии.
getProt	Возвращает запрос для получения определенного протокола

Мутации (см. Таблица 3):

Таблица 3

Наименование	Описание
lockRefFiles	Блокирует заданные файлы.
unlockRefFiles	Разблокирует заданные файлы.
recoverRefFileVersion	Восстанавливает версию файла.

Наименование	Описание
copyPrjRcRefFilesToPrjVisaSign	Копирование файлов проекта документа в Визу/Подпись.
putPrjRcToCabinetFolder	Положить проект документа в кабинеты.
markPrjFolderItemAsRead	Установить признак прочтения элемента папки.
markReadProt	Ставит отметку прочитанности на сущность.
returnPrjNum	Освобождает номер РКПД.

4.7.2 Кастомные аргументы

Список кастомных аргументов (см. Таблица 4):

Таблица 4

Наименование	Список мутаций	Описание
removeFromCabinet	updatePrjRc, updatePrjRc, updatePrjVisaSign, createPrjVisaSign	Удаление из кабинета
cascadeUpdateDeleted	updateRubricCl	Отметка логического удаления для дочерних записей справочника
classifJoin	updateCitizen	Признак объединенного сохранения записей справочника
contactJoin	updateOrganizCl	Признак объединения контактов
registerDocFromPrj	createDocRc	Регистрация РК из РКПД
useUserParms	createDocRc	Положить документ в папки, исключая
loggingForDeletedFiles	createDocRc	Протоколирование удалённых при регистрации РК файлов
rcSpecMode	updateDocRc	Спец. режимы при обновлении РК
currentCardCab	Все сущности DocRc и Resolution	Текущая картотека и кабинет.
noValidateMandatoryRules	createPrjRc, updatePrjRc, createDocRc, updateDocRc	Не проводить проверку правил.
rcSaveMode	updateDocRc	Строковый параметр для процедуры RC_SAVE
clearCtrlRes	createDocRc, updateDocRc	Строковый параметр для процедуры RC_SAVE
printReestr2Journal	updateJournal	Для установки номера и даты в реестр.
rcDeleteSpecMode	deleteDocRc	Спец. условия при удалении РК
rcInReg	updateDocRc, deleteDocRc	Режим "Регистрация РК"

Наименование	Список мутаций	Описание
refFileEds	deleteRefFileEds	Проверка закрытого ключа
prjApplyEds	updatePrjVisaSign	Зачистить "лишние" записи в разделе ЭП файлов в РКПД
prjApplyExecEds	updatePrjVisaSign	Зачистить "лишние" записи в разделе ЭП файлов в РКПД
useDeletedNumbers	deletePrjRc, deleteDocRc	Использовать номера удаляемых РКПД и РК
skipMandatoryRules	createPrjRc, updatePrjRc, createDocRc, updateDocRc	Не проверять определенные правила.
departmentCabinet	updateDepartment	Действия при изменении кабинета
departmentCascadeDates	createDepartment, updateDepartment	Обновить даты при добавлении или изменении
tempRcFileInfo	createTempRc, updateTempRc	Информация о группе документов и типе файлов для привязки

Для получения протокола реализован кастомный запрос `getProt`, который возвращает `IQueryable`, и, следовательно, генерирует страничный и курсорный пейджинг.

Параметры - `oper`, `ownerRefsns`:

Таблица 5

Значение параметра <code>oper</code>	Значение параметра <code>ownerRefsns</code>	Описание
P	Isn проекта документа	Протокол изменений РКПД
PEXE	Isn проекта документа	Протокол исполнителей РКПД
PSND	Isn проекта документа	Протокол адресатов РКПД
PTEM	Isn проекта документа	Протокол рубрик РКПД
PLNK	Isn проекта документа	Протокол связок РКПД
PARD	Isn проекта документа	Протокол доп. реквизитов РКПД
PVIS	Isn проекта документа	Протокол виз РКПД
PSGN	Isn проекта документа	Протокол подписей РКПД
PVS	Isn проекта документа	Протокол виз и подписей РКПД
PFIL	Isn проекта документа	Протокол файлов РКПД/виз/подписей

GetProtView

Для получения протокола реализован кастомный запрос `getProtView` (возвращает кастомную немапленную таблицу `ProtView`), который возвращает `IQueryable`, и следовательно генерирует страничный и курсорный пейджинг.

Параметры - `oper`, `inputs`

Значение параметра <code>oper</code>	Значение параметра <code>inputs</code>	Описание
PFIL	Пары: идентификатор объекта, тип объекта в виде строки {"Isn файла, 128"} {"Isn РКПД, 7"} {"Isn визы/подписи, 595"}	Протокол файлов РКПД/виз/подписей

Пример простого запроса:

```
// Запрос на получение протокола связок РКПД
{
  getProt(
    oper: "PLNK",
    ownerRefIsns: [4377, 4384])
  {
    operDescribe
    refIsn
    suboperId
    userIsn
    operComment
  }
}
```

Протокол `getProtView` на файлы:

```
// Запрос на получение протокола файлов
{
  getProtView(
    oper: "PFIL",
    inputs: ["501,128"])
  {
    refIsn
    operComment
    description
    note
    userIsn
    userName
  }
}
```

Пример запроса с пейджингом:

```
// Запрос на получение протокола связок РКПД с пейджингом
{
  getProtCr(
    oper: "PLNK",
    ownerRefIsns: [4377, 4384]
    filter: { and: [ { refIsn: { equal: { value: 11 } } }, { tableId: { equal:
{ value: ""К"" } } } ] }
    orderby: [{ operDescribe: Desc }])
  {
    items {
      operDescribe
      refIsn
      suboperId
      userIsn
      operComment
    }
  }
}
```

```
}  
}
```

В случае с пейджингом дополнительные фильтры устанавливаются поверх основного запроса.

5 C# Entity Framework (для ДЕЛО API 2023)

Актуальная технология: .NET Core, версия .NET 6.0.

Версия Entity Framework - 6.0.2

Контекст доступа к данным, а также любой сервис (сервисы кеширования, сервисы проверки прав пользователя) доступны через механизм DI (Dependency Injections).

5.1 Модель данных, используемая в API

Модель данных строится на следующих понятиях:

Атрибут – обычно соответствует полю таблицы БД.

Атрибут может быть одного из следующих типов:

Строка: формат String

Целое число: формат Int64, в БД может иметь формат Int16, Int32.

Дата-время: формат DateTime в UTC, в БД может иметь формат локального времени.

Дробное (десятичное) число: формат Decimal

Перечисление: тип enum

Коллекция сущностей: IEnumerable<T>, где T - это тип интерфейса сущности

Ссылка на сущность: тип интерфейса сущности

Интерфейс сущности - описывает структуру таблицы в БД. Все интерфейсы сущностей наследуются от интерфейса IKeyedEntity. Интерфейсы для сущностей генерируются на основе метаданных hbm.xml. Генерация происходит в момент сборки приложения. Интерфейсы могут наследоваться друг от друга.

Класс сущности – соответствует строке таблицы БД. Все сущности наследуются от соответствующих генерируемых интерфейсов. Класс сущности может реализовывать основной и несколько дополнительных интерфейсов (расширение класса). Генерация сущностей происходит в момент старта приложения.

Коллекция сущностей – объединяет множество сущностей одного типа. В коде представлена как интерфейс IEnumerable от соответствующего интерфейса сущности. Соответствует набору строк таблицы.

Ссылка на сущность – представляет идентификатор сущности. В коде представлена, как интерфейс соответствующей сущности.

Классы модели данных системы «ДЕЛО» определены в сборке Eos.Delo.Platform.Storage.Module, пространство имен - Eos.Delo.Platform.Storage.Model.

Рассмотрим описанные понятия на примере некоторых сущностей системы.

Пример кода интерфейса сущности:

```
// Интерфейс сущности, соответствующей таблице DOC_RC
public interface IDocRc : IKeyedEntity
{
    Int64? IsnDoc
    {
        get;
        set;
    }

    Int64 KindDoc
    {
        get;
        set;
    }
}
```

```

String DueDocgroup
{
    get;
    set;
}

DateTime DocDate
{
    get;
    set;
}

...

IEnumerable<IAcquaintance> Acquaintances
{
    get;
    set;
}

IEnumerable<IDocExe> DocExes
{
    get;
    set;
}

IEnumerable<IDocFolderItem> DocFolderItems
{
    get;
    set;
}

...

ISecurityCl SecurityCl
{
    get;
    set;
}

INomenklCl NomenklCl
{
    get;
    set;
}

IDocvidCl DocvidCl
{
    get;
    set;
}

IArRcValue ArRcValue
{
    get;
    set;
}

IQueryable<IRefLink> GetRefLinks(IDataContext context);
IQueryable<IRefFile> GetRefFiles(IDataContext context);
}

```

Интерфейс IDocRc является интерфейсом сущности, соответствующей таблице DOC_RC. Атрибутами этого объекта являются, например, свойства DueDocgroup и IsnDoc, которые соответствуют полям таблицы - DUE_DOCGROUP и ISN_DOC.

Поле Department типа IDepartment является ссылкой на сущность. Интерфейс IDepartment соответствует таблице DEPARTMENT. Поле Resolutions типа IEnumerable<IResolution> является коллекцией сущностей для интерфейса сущности.

Все интерфейсы сущностей унаследованы от базового интерфейса IKeyedEntity, который имеет полезное свойство-индексатор, позволяющее обращаться к элементам сущности по имени в случаях, когда конкретный тип или элементы типа не известны на момент разработки.

Пример кода описания базового интерфейса, от которого наследуются все сущности:

```
// Описание базового интерфейса, от которого наследуются все сущности
public interface IKeyedEntity
{
    object[] KeyValues { get; }

    [System.Runtime.CompilerServices.IndexerName("__Item")]
    object this[string name] { get; set; }

    ...
}
```

В процессе разработки прикладного кода доступны только интерфейсы сущностей. Классы, реализующие интерфейсы и позволяющие работать с БД посредством ORM Entity Framework не доступны для непосредственного использования. Код классов генерируется динамически при старте приложения.

Пример динамической генерации кода классов:

```
// Динамическая генерация кода класса
[Display(Name="DocRc_name", Description="DocRc_description",
ResourceType=typeof(Eos.Delo.Platform.Storage.Properties.Resources))]
public partial class DocRc : IDocRc
{

    partial void Initialize();

    private Int32? _IsnDoc;

    private Int16 _KindDoc;

    private String _DueDocgroup;

    ...

    private ICollection<Acquaintance> _Acquaintances;

    private ICollection<DocExe> _DocExes;
```

```

private ICollection<DocFolderItem> _DocFolderItems;

...

private SecurityCl _SecurityCl;

private NomenklCl _NomenklCl;

private DocvidCl _DocvidCl;

private ArRcValue _ArRcValue;

internal DocRc()
{
    Initialize();
    this._Acquaintances = new HashSet<Acquaintance>();
    this._DocExes = new HashSet<DocExe>();
    this._DocFolderItems = new HashSet<DocFolderItem>();
}

[NotMapped()]
public virtual object [] KeyValues
{
    get
    {
        return new object [] { IsnDoc };
    }
}

[Display(Name="DocRc_IsnDoc_name",
Description="DocRc_IsnDoc_description",
ResourceType=typeof(Eos.Delo.Platform.Storage.Properties.Resources))]
[Range(-2147483648, 2147483647, ErrorMessageResourceName="range",
ErrorMessageResourceType=typeof(Eos.Delo.Platform.Storage.Properties.Resources
))]

public virtual Int32? IsnDoc
{
    get
    {
        return this._IsnDoc;
    }
    set
    {
        this._IsnDoc = value;
    }
}

```

```

Int64? IDocRc.IsnDoc
{
    get
    {
        return this.IsnDoc;
    }
    set
    {
        checked
        {
            this.IsnDoc = (Int32?)value;
        }
    }
}

[System.Runtime.CompilerServices.IndexerName("__Item")]
public virtual object this[string name]
{
    get
    {
        if ((name == "IsnDoc"))
        {
            return this.IsnDoc;
        }
    }
}
}

```

В модели данных прикладной системы ряд интерфейсов сущностей наследуются от общего базового интерфейса.

Также некоторые сущности в системе имеют возможность расширения - т.е. добавление реквизитов, не включенных в стандартную модель системы. С помощью этого механизма реализованы дополнительные реквизиты РК, РКПД и некоторых справочников.

На пример кода класса сущности дополнительных реквизитов РКПД, видно, что класс реализует два интерфейса: IArPrjValue, IArPrjValueAdd. Первый является базовым, и входит в базовую модель, в то время как второй формируется динамически в процессе работы системы по мере добавления новых реквизитов через приложение Администрирования.

Пример кода класса сущности с дополнительными реквизитами:

```

// Класс сущности РКПД с дополнительными реквизитами: IArPrjValue - базовый,
IArPrjValueAdd - формируется динамически
public partial class ArPrjValue : IArPrjValue, IArPrjValueAdd
{

    partial void Initialize();

    private String _EosItemNumber;

```

```

...

private String _MedoDocKind;

private Int32? _IsnPrj;

private PrjRc _PrjRc;

internal ArPrjValue()
{
    Initialize();
}

[NotMapped()]
public virtual object [] KeyValues
{
    get
    {
        return new object [] { IsnPrj };
    }
}

[Display(Name="ArPrjValue_EosItemNumber_name",
Description="ArPrjValue_EosItemNumber_description",
ResourceType=typeof(Eos.Delo.Platform.Storage.Properties.Resources))]
[MaxLength(255, ErrorMessageResourceName="maxlength",
ErrorMessageResourceType=typeof(Eos.Delo.Platform.Storage.Properties.Resources
))]

public virtual String EosItemNumber
{
    get
    {
        return this._EosItemNumber;
    }
    set
    {
        this._EosItemNumber = value;
    }
}
}

```

Полный перечень определенных в API сущностей системы и их связей приведен в Приложение Г, Приложение Д.

5.2 Внедрение зависимостей (DI)

DI - один из основных принципов работы в современных приложениях на .NET Core. DI позволяет широко использовать собственные сервисы. Более подробная информация:

<https://docs.microsoft.com/en-us/aspnet/core/fundamentals/dependency-injection?view=aspnetcore-6.0>

Пример регистрации собственного сервиса:

```
// Описание класса с реализацией собственного сервиса
public class SampleService
{
    private readonly IDataContext _dataContext;
    private readonly ILogger _logger;

    public SampleService(IDataContext dataContext, ILogger<SampleService>
logger)
    {
        _dataContext = dataContext;
        _logger = logger;
    }

    public void SomeMethod()
    {
        _logger.LogInformation("Run SomeMethod");
    }
}

// Регистрация сервиса в методе RegisterServices:
services.AddScoped<SampleService>();

// Использование сервиса:
private readonly SampleService _sampleService;

public CustomQueriesSample(SampleService sampleService)
{
    _sampleService = sampleService;
}

...

private void CallSomeMethod()
{
    _sampleService.SomeMethod();
}
```

5.3 Модули платформы

Для использования основных возможностей платформы нужно подключить в свой проект модуль платформы - **Eos.Platform.Storage.Module**.

Для использования основных возможностей и сервисов системы «ДЕЛО» нужно подключить в свой проект модуль - **Eos.Delo.Platform.Storage.Module**.

5.4 Контекст доступа к данным - IDataContext

Модуль - Eos.Delo.Platform.Storage.Module;

Пространство имен - Eos.Delo.Platform.Storage.Model.

Доступ к данным осуществляется с использованием Entity Framework через интерфейс **IDataContext** (наследуется от DbContext). Является scoped-сервисом.

Так как IDataContext является scoped-сервисом, он существует в рамках запроса, либо его нужно создавать через ServiceScopeFactory как локальную переменную, с использованием using для создания скопа. Нельзя держать IDataContext внутри singleton-сервиса как внутреннюю переменную, так как остается открытым соединение.

Рекомендации по работе с IDataContext:

- не нужно IDataContext хранить в свойствах/полях долгоживущих классов.
- т.к. IDataContext получаем из ServiceProvider, то за время его жизни отвечает тоже ServiceProvider - не нужно его брать в using и не нужно его явно вызывать Dispose().
- в ФЗ нужно внимательно следить, чтобы случайно не начать работать с корневым ServiceProvider-ом, обязательно создавать ServiceScope, и именно его брать в using или закрывать явным Dispose().

Ниже приведен пример получения и сохранения данных через IDataContext с использованием стандартных средств Entity Framework (**Не рекомендуется пользоваться стандартными средствами, вместо этого используйте перегруженные методы (см. 5.4.1. Конвейер промежуточного ПО), чтобы запустить конвейер обработки и выполнить необходимые проверки и алгоритмы**)

* **Примечание:** для изменения данных не рекомендуется использовать метод Update, так как он обновляет все поля в БД. Вместо этого используйте **EnsureTracked** из Eos.Delo.Platform.Storage.Exctensions, затем поменяйте поля и вызовите метод SaveChangesAsync.

* **Примечание:** для получения данных желательно использовать **AsNoTracking**, чтобы данные не попадали в ChangeTracker.

* **Примечание:** методы Add и Remove вызывают каскадное добавление и удаление сущностей, если к добавляемой сущности привязать другие сущности через навигационные свойства. Для того, чтобы избежать такого поведения используйте методы **SetAddedState** и **SetDeletedState** из Eos.Delo.Platform.Storage.Exctensions.

Пример получения и сохранения данных через IDataContext:

```
// Получение и сохранение данных через IDataContext
using Eos.Delo.Platform.Storage.Exctensions;
using Eos.Delo.Platform.Storage.Model;

private readonly IDataContext _dataContext;
private readonly DataContextHelperService _dataContextHelper;

public MyClass(IDataContext dataContext, DataContextHelperService
dataContextHelper)
{
    _dataContext = dataContext;
    _dataContextHelper = dataContextHelper;
}

public async Task SimpleAddressQueryAsync (MiddlewareContextBase
baseContext)
{
    var address = await _dataContext.Addresses.Where (b => b.IsnAddress
== 1)
        .AsNoTracking ()
```



```

        .FirstOrDefaultAsync();

        var address2 = await _dataContext.Addresses.Where(b =>
b.IsnAddress == 2)
        .AsNoTracking()
        .FirstOrDefaultAsync();

        var address3 = await
_dataContextHelper.GetEntityByPrimaryKeyAsync(3);

        _dataContext.EnsureTracked(address3);
        address3.Note = "Тест3";

        _dataContext.EnsureTracked(address2);
        address2.Note = "Тест1";

        var citizen = await _dataContext.Citizens.Where(p => p.IsnCitizen
== 3935)
        .AsNoTracking()
        .FirstOrDefaultAsync();

        var newAddress = _dataContext.CreateAddress();
        newAddress.IsnRegion = 4057229;
        newAddress.Note = "Тест";
        newAddress.IsnOwner = citizen.IsnCitizen;
        newAddress.KindOwner = 109;
        _dataContext.SetAddedState(newAddress);
        await _dataContext.SaveChangesAsync();

        _dataContext.SetDeletedState(address);
        await _dataContext.SaveChangesAsync();
    }

```

5.4.1 Конвейер промежуточного ПО

Одной из основных возможностей платформенного IDataContext является прогон изменений через конвейер (pipeline) обработки промежуточного ПО (middleware). Это позволяет имитировать работу триггеров на app-сервере. На прямом проходе производятся различные проверки прав и установка значений по умолчанию. На обратном проходе могут вызываться различные алгоритмы.

Для сохранения данных с выполнением бизнес-логики в конвейере используется перегруженный метод `SaveChangesAsync` из пространства имен `Eos.Platform.Storage.Middleware`.

В метод `SaveChangesAsync` необходимо передать `ServiceProvider` и `MutationContext`. `MutationContext` можно создавать используя `MiddlewareHelper.CreateMutationContext` из `Eos.Delo.Platform.Storage.Helpers`. `MutationContext` для каждого нового захода через pipeline должен быть новым. Не используйте `MutationContext` как глобальную переменную!

Для чтения с проверкой прав доступа в конвейере используется один из перегруженных методов `FirstOrDefaultAsync`, `AnyAsync` и т.д. из пространства имен - `Eos.Platform.Storage.Middleware`.

Примеры использования:

1) Добавление:

```
// Конвейер промежуточного ПО - добавление
using Eos.Platform.Storage.Middleware;
using Eos.Delo.Platform.Storage.Helpers;
using Eos.Delo.Platform.Storage.Model;
using Eos.Delo.Platform.Storage.Extensions;

private readonly IDataContext _dataContext;
private readonly IServiceProvider _serviceProvider;

public MyClass(IDataContext db, IServiceProvider serviceProvider)
{
    _serviceProvider = serviceProvider;
    _dataContext = db;
}

public async Task CreateTestAddressAsync(MiddlewareContextBase
baseContext)
{
    Random rnd = new Random(DateTime.UtcNow.Second);

    var newAddress = _dataContext.CreateAddress();
    newAddress.IsnRegion = 4057225;
    newAddress.Note = "Тест" + RandomString(rnd, 10);
    newAddress.IsnOwner = 3935;
    newAddress.KindOwner = 109;

    var currentUser = baseContext.GetDeloContext().User;

    var context = MiddlewareHelper.CreateMutationContext(_dataContext,
currentUser, null,
ExtProtOperationType.Mutation, baseContext.CancellationToken);

    _dataContext.SetAddedState(newAddress);

    await _dataContext.SaveChangesAsync(_serviceProvider, context);
}
```

2) Изменение:

```
// Конвейер промежуточного ПО - изменение
using Eos.Platform.Storage.Middleware;
using Eos.Delo.Platform.Storage.Helpers;
using Eos.Delo.Platform.Storage.Model;
```

```

using Eos.Delo.Platform.Storage.Extensions;

private readonly IDataContext _dataContext;
private readonly IServiceProvider _serviceProvider;

public MyClass(IDataContext db, IServiceProvider serviceProvider)
{
    _serviceProvider = serviceProvider;
    _dataContext = db;
}

public async Task UpdateTestAddressAsync(long isnAddress,
MiddlewareContextBase baseContext)
{
    Random rnd = new Random(DateTime.UtcNow.Second);

    var address = await _dataContext.Addresses.Where(b => b.IsnAddress
== isnAddress)
        .AsNoTracking()
        .FirstOrDefaultAsync(_serviceProvider, new QueryContext<IAddre
ss>(baseContext));

    _dataContext.EnsureTracked(address);
    address.Note = "Тест" + RandomString(rnd, 10);

    var currentUser = baseContext.GetDeloContext().User;

    var context = MiddlewareHelper.CreateMutationContext(_dataContext,
currentUser, null,
        ExtProtOperationType.Mutation, baseContext.CancellationToken);

    await _dataContext.SaveChangesAsync(_serviceProvider, context);
}

```

3) Удаление:

```

// Конвейер промежуточного ПО - удаление
using Eos.Platform.Storage.Middleware;
using Eos.Delo.Platform.Storage.Helpers;
using Eos.Delo.Platform.Storage.Model;
using Eos.Delo.Platform.Storage.Extensions;

private readonly IDataContext _dataContext;
private readonly IServiceProvider _serviceProvider;

public MyClass(IDataContext db, IServiceProvider serviceProvider)
{
    _serviceProvider = serviceProvider;
}

```

```

        _dataContext = db;
    }

    public async Task DeleteTestAddressAsync(long isnAddress,
        MiddlewareContextBase baseContext)
    {
        var address = await _dataContext.Addresses.Where(b => b.IsnAddress
            == isnAddress)
            .AsNoTracking()
            .FirstOrDefaultAsync(_serviceProvider, new QueryContext<IAddress>(baseContext));

        _dataContext.SetDeletedState(address);

        var currentUser = baseContext.GetDeloContext().User;

        var context = MiddlewareHelper.CreateMutationContext(_dataContext,
            currentUser, null,
            ExtProtOperationType.Mutation, baseContext.CancellationToken);

        await _dataContext.SaveChangesAsync(_serviceProvider, context);
    }

```

5.5 Сервис DataContextHelperService

Модуль - Eos.Delo.Platform.Storage.Module;

Пространство имен - Eos.Delo.Platform.Storage.Services;

Расширения - Eos.Delo.Platform.Storage.Extensions.

Сервис **DataContextHelperService** позволяет быстро получить необходимую сущность по первичному, альтернативному ключу, навигационному свойству или по ряду условий, выполнять хранимые процедуры или SQL-код, получать сервисы для реализации бизнес-логики или проверки прав пользователя на РК, РКПД, Резолюции. Также позволяет получать коллекции родительского объекта через этот объект, либо целые таблицы. Кроме того, позволяет получать коллекцию через UserLists.

При получении коллекции, если в родительской сущности уже заполнен список для этой коллекции (неважно, частично или полностью), каждая сущность в списке обновит свои значения на актуальные из БД.

В Extensions содержатся методы определения дубликата записи по определенным параметрам. Также содержатся методы получения определенных коллекций, методы вызова определенных процедур, методы получения коллекций через UserLists, методы получения сущностей и реализации более сложной логики, например GetPareRefLinkAsync получает парную связку через RefLink.

Сервис DataContextHelperService необходимо использовать для получения классов, которые используются для проверки прав и для вспомогательных методов (например добавления исполнителя к РКПД и т. д.). Примеры классов: PrjRcSecurityService, PrjRcHelperService и другие. Полное описание есть в списке методов для DataContextHelperService.

В актуальной версии работа через DataContextHelperService является стандартом работы в системе и при использовании подключаемых модулей.

Примеры использования:

1) Получение сущности по первичному ключу:

```
// Получение сущности по первичному ключу
using Eos.Delo.Platform.Storage.Services;

private readonly DataContextHelperService _dch;

public MyClass(DataContextHelperService dch)
{
    _dch = dch;
}

public async Task SomeMethod(CancellationTokens cancellationTokens = default)
{
    var prjRc = await _dch.GetEntityByPrimarykeyAsync<IPrjRc>(1,
cancellationTokens);
}
```

2) Получение сущности по ключу:

```
// Получение сущности по ключу
using Eos.Delo.Platform.Storage.Services;

private readonly DataContextHelperService _dch;

public MyClass(DataContextHelperService dch)
{
    _dch = dch;
}

public async Task SomeMethod(CancellationTokens cancellationTokens = default)
{
    var department = await
_dch.GetEntityByKeyAsync<IDepartment>(nameof(IDepartment.IsnNode), 1,
cancellationTokens);
}
```

3) Получение сущности через навигационное свойство:

```
// Получение сущности через навигационное устройство
using Eos.Delo.Platform.Storage.Services;
using Eos.Delo.Platform.Storage.Extensions;

private readonly DataContextHelperService _dch;

public MyClass(DataContextHelperService dch)
{
    _dch = dch;
}
```

```
public async Task SomeMethod(CancellationTokен cancellationTokен = default)
{
    var department = await _dch.GetEntityByPrimaryKeyAsync<IDepartment>(1,
cancellationTokен);
    var parentDep = await _dch.GetEntityFromReferenceAsync(department, p =>
p.ParentNode, cancellationTokен);
}
```

4) Получение коллекции через объект:

```
// Получение коллекции через объект
using Eos.Delo.Platform.Storage.Services;

private readonly DataContextHelperService _dch;

public MyClass(DataContextHelperService dch)
{
    _dch = dch;
}

public async Task SomeMethod(CancellationTokен cancellationTokен = default)
{
    var user = await _dch.GetEntityByPrimaryKeyAsync<IUserCl>(1,
cancellationTokен);
    var userCardList = await _dch.GetCollectionAsync(user, p => p.UserCards,
cancellationTokен, CacheUse.Normal);

    var prjRc = await _dch.GetEntityByPrimaryKeyAsync<IPrjRc>(1,
cancellationTokен);
    var prjExecs = (await _dch.GetCollectionAsync(prjRc, p => p.PrjExecs,
cancellationTokен)).Where(p => p.CanManageExec == 1).ToList();
}
```

Для получения полиморфных коллекций можно использовать "родные" методы DataContextHelperService или готовые методы расширений.

Пример получения списка файлов:

```
// Получение списка файлов
public static async Task<IEnumerable<IRefFile>>
GetRefFileListAsync(this DataContextHelperService dbHelper, IKeyedEntity
obj, string parentColumnName, long refFileType, CancellationTokен
cancellationTokен = default, CacheUse cacheUse = CacheUse.Normal) =>
    await
dbHelper.GetCollectionAsync<IRefFile>(nameof(IRefFile.IsnObject),
nameof(IRefFile.Owner), parentColumnName, obj, false, cancellationTokен,
cacheUse, nameof(IRefFile.RefFileType), refFileType);
```

Все сущности и коллекции через DataContextHelperService загружаются AsNoTracking.

Описание методов приведено в Приложение Р.

Пример получения сервисов проверки прав:

```
// Получение сервисов проверки прав
using Eos.Delo.Platform.Storage.Services;
using Eos.Delo.Platform.Storage.Extensions;

private readonly DataContextHelperService _dch;

public MyClass(DataContextHelperService dch, IDataContext db)
{
    _dch = dch;
}

public async Task SomeMethod(MiddlewareContextBase baseContext)
{
    _dch.Init(baseContext.GetDeloContext().User);
    var userRightsHelperService = _dch.UserRightsHelper();
    var prjRc = await _dch.GetEntityByPrimaryKeyAsync<IPrjRc>(1,
cancellationTokens);
    var prjRcSecurityService = _dch.PrjRcSecurity(PrjRc);
}
```

5.6 Права пользователя

5.6.1 Статический класс расширения UserExtensions

Модуль - Eos.Delo.Platform.Storage.Module;

Пространство имен - Eos.Delo.Platform.Storage.Extensions.

В классе UserExtensions определены методы для проверки, есть ли у пользователя абсолютное право или право системного технолога.

Пример проверки прав пользователя:

```
// Проверка прав пользователей
private readonly UserCachingService _userService;

public Constructor(UserCachingService userService)
{
    _userService = userService;
}

public async Task SomeMethodAsync(MutationContext context)
{
    var user = await _userService.GetCurrentUserAsync(context.User);

    var hasRight = User.HasDeloRight(AbsoluteRight.Admin);
}
```

Описание публичных методов находится в Приложение Н

5.7 Сервисы кэширования

Для кэширования, группировки и систематизации различных объектов базы данных были разработаны сервисы кэширования.

Модуль - Eos.Delo.Platform.Storage.Module;
 Пространство имен - Eos.Delo.Platform.Storage.Services;
 Сервисы кэширования - Eos.Delo.Platform.Storage.Services.Caching;
 Объекты кэширования –
 Eos.Delo.Platform.Storage.Services.Caching.Model;
 Фоновые задачи –
 Eos.Delo.Platform.Storage.Services.Caching.Daemons.

Пример использования сервисов кэширования:

```
// Использование сервисов кэширования
private readonly SysParamsCachingService _sysParamsService;

public Constructor(SysParamsCachingService sysParamsService)
{
    _sysParamsService = sysParamsService;
}

var sysParams = await _sysParamsService.GetOrAddAsync(cancellationToken);
var passwordDate = sysParams.PassDate != 0 ?
DateTime.UtcNow.AddDays(sysParams.PassDate) : (DateTime?)null;
```

Список доступных сервисов кэширования в Приложение П

5.8 Пакетная модификация в базе данных

Модуль - Eos.Platform.Storage.Module;

Пространство имен – **Thinktecture**.

Платформенный Thinktecture - это платформенная реализация библиотеки Thinktecture, которая позволяет производить групповые операции с данными.

*** Примечание:** полезные функции, такие, как LIKE находятся в стандартных методах расширения **EF.Functions**.

Пример использования пакетной модификации в базе данных:

```
// Пакетная модификация в базе данных
using Microsoft.EntityFrameworkCore;
using Thinktecture;

public async Task SomeMethod()
{
    await _db.Addresses.Where(p => EF.Functions.Like("тест", p.Note
+ "%")).BulkDeleteAsync();
}
```

5.9 Работа с файлами

Пример работы с файлами через API:

```
// пример работы с файлами через API
[Api(RequestType.Mutation, FieldName = "sampleCopyFile", Description =
"Пример копирования файла")]
public async Task RefFile_APICopySuccess(LongLongArgument arg, Middlew
areContextBase baseContext)
{
    var context = MiddlewareHelper.CreateMutationContext(baseContext);
```



```

var refFile = await _dataContext.RefFiles
    .Where(p => p.IsnRefFile == arg.IdFrom)
    .FirstOrDefaultAsync();

if (refFile != null)
{
    IRefFile refFileForAdd = _dataContext.CreateRefFile();

    refFileForAdd.IsnObject = arg.IdTo;
    refFileForAdd.RefFileType = RefFileType.RefFileVisaSign;
    refFileForAdd.AccessMode = refFile.AccessMode;
    refFileForAdd.Securlevel = refFile.Securlevel;
    refFileForAdd.IsnFileType = refFile.IsnFileType;
    refFileForAdd.Tag = refFile.Tag;
    refFileForAdd.EpguFlag = refFile.EpguFlag;
    refFileForAdd.IsHidden = refFile.IsHidden;
    refFileForAdd.DontdelFlag = refFile.DontdelFlag;
    refFileForAdd.SendEnabled = refFile.SendEnabled;
    refFileForAdd.ApplyEds = refFile.ApplyEds;

    _dataContext.SetAddedState(refFileForAdd);

    refFileForAdd.Contents.SetSource(refFile.Contents, context);
}

await _dataContext.SaveChangesAsync(_serviceProvider, context);
}

```

Также примеры работы с файлами через API расположены в дистрибутиве системы в каталоге «Sdk» (CustomQueriesSample.cs).

5.10 Логирование

Для логирования используется интерфейс ILogger типизированный искомым классом. ILogger можно достать через DI.

Пример логирования:

```

// Логирование
public class SampleLogging
{
    private readonly ILogger _logger;

    public SampleLogging(ILogger<SampleLogging> logger)
    {
        _logger = logger;
    }

    public void SomeMethod()
    {
        _logger.LogInformation("Run SomeMethod");
    }
}

```

6 Фоновые задачи

6.1 Введение в API фоновых задач

Сервер фоновых задач (хост) предоставляет возможность выполнять и контролировать ход выполнения фоновых задач для системы «ДЕЛО».

Фоновые задачи (ФЗ) предназначены для решения таких вопросов, как автоматизация маршрутизации и обработка документов, интеграция со сторонними системами, оповещение пользователей о событиях в системе по электронной почте и др.

Фоновые задачи выполняются на хосте, входящем в состав серверной части системы «ДЕЛО». Для того, чтобы сервер фоновых задач начал выполнение фоновой задачи, библиотека, содержащая фоновую задачу, должна быть предварительно загружена на хост и сконфигурирована. После этого, сервер фоновых задач может начать выполнение задач, содержащихся в загруженных библиотеках, автоматически, либо по запросу пользователя через WEB-интерфейс управления фоновыми задачами системы «ДЕЛО».

Для разработки фоновых задач с использованием API фоновых задач, разработчик должен владеть технологией .NET Core. API фоновых задач предоставляет следующие возможности:

- Работать с объектами БД с использованием строго типизированной модели данных.
- Выполнять поиск объектов в БД по заданным критериям.
- Создавать и модифицировать объекты в БД.
- Получать оповещение о событиях создания объектов БД и изменения их состояния.
- Работать с контекстом выполняющейся фоновой задачи.
- Работать с файлами.

Все перечисленные возможности предоставляются через специализированные интерфейсы и методы классов, **IDataContext** (5.4. Контекст доступа к данным - IDataContext), для взаимодействия с БД, получения, добавления, изменения и удаления сущностей, сервис **EntityHelperService** с методами для поиска и загрузки сущностей системы и др.

6.2 Модель данных, используемая в API фоновых задач

Модель данных описана в 5.1. Модель данных, используемая в API.

Пример получения данных показан в описании **IDataContext** (5.4. Контекст доступа к данным - IDataContext).

6.3 Разработка фоновой задачи (для ДЕЛО API 2009 и ДЕЛО API 2023)

Взаимодействие с системой «ДЕЛО» из фоновых задач осуществляется с помощью интерфейса **IDataContext** и сервисов, **EntityHelperService**, сервисов кеширования. Эти интерфейсы и классы описаны в разделе 5. C# Entity Framework (для ДЕЛО API 2023).

Разработка новых проектов на ДЕЛО API 2009 не осуществляется. Необходимые изменения для поддержки работы старых проектов на ДЕЛО API 2009 описаны в текущем разделе в пунктах 6.3.1. Создание проекта, 6.3.2. Создание класса фоновой задачи, наследника **BaseEventDispatcher** / **BaseDispatcher** / **BaseDaemon**, 6.3.3 Чтение конфигурации фоновой задачи. Пункт 6.4. Подготовка фоновой задачи к выполнению одинаков для ДЕЛО API 2023 и ДЕЛО API 2009. ДЕЛО API 2009 описано в руководстве программиста предыдущей версии.

6.3.1 Создание проекта

В начале разработки, в Visual Studio необходимо создать проект .NET Core Class Library, версия framework - .NET 6.0. Для проектов на ДЕЛО API 2009 также необходимо поднять версию framework до .NET 6.0.

Далее, добавить в проект NuGet пакет Eos.Delo.Daemon.Dispatchers, он также содержит ссылки на NuGet пакеты Eos.Delo.Platform.Storage.Module, Eos.Platform.Daemon, Eos.Platform.Host.Module необходимые в разработке.

Eos.Delo.Platform.Storage.Module содержит базовые классы и интерфейсы, например, IDataContext для доступа к данным, EntityHelperService для получения сущностей и др.

Eos.Platform.Storage.Module содержит основную платформенную функциональность.

Eos.Platform.Daemon содержит базовый класс фоновых задач BaseDaemon, базовый платформенный диспетчер BaseDispatcher, методы для работы хоста.

Для проектов на ДЕЛО API 2009 необходимо подключить NuGet пакет Eos.Delo.Api.Module содержащий ссылки на NuGet пакеты Eos.Delo.Common, Eos.Delo.Utills, Eos.Delo96.Api и др.

6.3.2 Создание класса фоновой задачи, наследника BaseEventDispatcher / BaseDispatcher / BaseDaemon

При разработке фоновой задачи необходимо создать базовый класс ФЗ по определённому шаблону.

Класс ФЗ наследует один из классов BaseEventDispatcher / BaseDispatcher (Eos.Delo.Daemon.Dispatchers) или BaseDaemon (Eos.Platform.Daemon.Types) при невозможности реализации через базовые диспетчеры. Для реализации фоновых задач на платформе в большинстве случаев подойдет BaseDispatcher (Eos.Delo.Daemon.Dispatchers), это стандартизированный шаблон диспетчера, реализующий основной функционал для работы ФЗ.

Внимание!

Рекомендована реализация ФЗ на BaseEventDispatcher / BaseDispatcher. Реализация на BaseDaemon осуществляется в крайнем случае, если невозможно написание на одном из диспетчеров.

Дальнейшая реализация класса зависит от наследуемого класса.

Реализация на базовых диспетчерах:

1) BaseDispatcher (Eos.Delo.Daemon.Dispatchers) – общий диспетчер Дело для фоновых задач, который реализует в себе стандартный набор функций для работы с ватермаркой и параметрами. Для работы с ним необходимо его наследовать и реализовать методы FillParamsAsync и WorkAsync.

В метод FillParamsAsync передается экземпляр конфигурации и логика его заполнения: **Params = GetDefaultAppSettingsParams<TestDispatcherCfg>();** для общих настроек конфигурации и **JobParams = GetJobAppSettingsParams<TestDiapatcherJobCfg>();** для экземплярных.

Метод WorkAsync реализует необходимую обработку. Если необходимо соединение с базой данных, выполняемые конструкции пишутся в **InContextAsync((context) => { //action });**

Необходимо выполнять проверку полученных параметров на null.

Если отсутствует конфигурация данной ФЗ, необходимо сгенерировать исключение NoDaemonConfigException(). ФЗ будет остановлена с ошибкой.

Если конфигурация ФЗ некорректна, исключение InvalidDaemonConfigException (string message) или InvalidDaemonConfigException(IEnumerable<string> messages) с текстами ошибок. ФЗ будет остановлена с ошибкой.

Если отсутствует конфигурация другой ФЗ, конфигурация которой обязательна для данной, исключение `NoParentDaemonConfigException(string parentDaemonName)` с указанием названия ФЗ донора настроек. ФЗ выйдет из текущей итерации работы без статуса остановки с ошибкой.

Если при отсутствии корректной конфигурации требуется выйти из текущей итерации без остановки ФЗ с ошибкой, необходимо сгенерировать исключение `NoConfigurationException(string message)`.

2) `BaseEventDispatcher` - обертка над `BaseDispatcher`, которая стандартизирует работу ФЗ, работающих на событиях.

Для работы с этим диспетчером необходимо наследовать класс `BaseEventDispatcher` и реализовать методы `AgregateAsync`, `EventProcessAsync` и `FillParamsAsync`.

Метод `AgregateAsync` отвечает за отбор событий из списка. Метод получает n-кол-во событий, из которых необходимо отобрать нужные (агрегирование по флагам) и вернуть их списком.

Метод `EventProcessAsync` описывает основную логику работы с событием. В нем загружается все необходимое, производится валидация, и дальнейшая обработка события производится в обрабатывающем хелпере.

У обоих диспетчеров так же есть следующие методы и поля, которые можно переопределить:

- **InitAsync** - метод отвечает за инициализацию фоновой задачи. По умолчанию в нем установлен интервал запуска итерации ФЗ в 30 секунд.

- **FinishAsync** - метод отвечает за окончание итерации работы фоновой задачи. По умолчанию в нем вызывается метод `WaitAsync` с вычислением времени следующего запуска и ожиданием.

- **InContextAsync** - метод создает контекст соединения с БД, внутри которого выполняется передаваемый action. (Работа в контексте соединения с БД. Переопределяется в случае, если необходимо использовать DAPI с открытием фасада.)

- **SetWatermarkId** - метод позволяет установить ключ ватермарки. Не рекомендуется менять.

- **CS_WATERMARK_ID** - поле содержит VALUE_ID ватермарки.

`BaseEventDispatcher` так же имеет следующие свойства и методы:

SIZE - свойство отвечает за размер пейджинга при загрузке событий (максимальное количество загружаемых событий за раз).

UseDefaultStatistic - свойство отвечает за использование статистики по умолчанию (по умолчанию включено).

GetNextEventsAsync - метод отвечает за загрузку событий.

SaveStatsAsync - метод сохранения статистики по событию.

Диспетчер `BaseDispatcher` имеет метод `WorkAsync`, который необходимо реализовать в случае, если вы используете данный диспетчер, а не `BaseEventDispatcher`.

Для использования `BaseDispatcher` в своем проекте необходимо загрузить NuGet пакет `Eos.Delo.Daemon.Dispatchers`.

Для ДЕЛО API 2009 при наследовании `BaseEventDispatcher` / `BaseDispatcher` необходимо переопределить метод `InContextAsync(Func<Task> action)`. Данный контекст не поддерживает многопоточности, поэтому вызовы любых асинхронных методов в теле метода должны получать результат через `*.Result` или ожидание `*.Wait()`, использование `await` приведет к ошибкам.

Пример переопределения `InContextAsync` для ДЕЛО API 2009:

```
protected override async Task InContextAsync(Func<Task> action)
{
    try
```

```

{
    Logger.LogInformation("Начало InContextAsync");

    if (WorkContext.Current.Head != null)
        WorkContext.DropCurrent();
    var user = GetUserAsync().Result;
    Facade.Open(String.Empty);
    Facade.SetUserIsn(user.UserId);

    action();
}
catch (DaemonCancelException)
{
    throw; //Необходимо для остановки ФЗ
}
catch (Exception ex)
{
    MarkInfoError(ex);
    Logger.LogError(ex.ToString());
}
finally
{
    WorkContext.DropCurrent();
    Logger.LogInformation("Конец InContextAsync");
}
}

```

Примеры ФЗ, реализованные на диспетчерах, расположены в дистрибутиве системы в каталоге «Sdk» (проект Eos.Delo.Samples.Daemons).

Пример реализации ФЗ на BaseDispatcher:

```

// Реализация фоновой задачи на BaseDispatcher
public class MailSenderDispatcher : BaseDispatcher
{
    MailSenderCfg _params;

    protected override async Task InitAsync()
    {
        Delay.Minutes = 1;
    }

    protected override async Task FillParamsAsync()
    {
        _params = GetDefaultAppSettingsParams<MailSenderCfg>();
        if (_params == null)
            throw new NoDaemonConfigurationException();
    }

    protected override async Task WorkAsync()
    {
        var successSend = false;
        await InContextAsync(async (context, serviceProvider) => {
            var channel = serviceProvider.GetRequiredService<IEmailChannel>();
            var appSettingsService =
serviceProvider.GetRequiredService<IAppSettingsService>();
            var secretStore =
serviceProvider.GetRequiredService<ISecretStore>();
            successSend = channel.Send(_params.EmailCfgInstanceId,
appSettingsService, secretStore, _params.Admin, $"Тестовое тело письма\nIs
Linux = {System.OperatingSystem.IsLinux()}\"", Logger);
        });

        if (!successSend)
            throw new Exception("Ошибка отправки email"); }
}

```

```
}
```

Пример реализации ФЗ на BaseEventDispatcher:

```
// Реализация фоновой задачи на BaseEventDispatcher
public class AddresseeDirectionDispatcher : BaseEventDispatcher
{
    private AddresseeDirectionJobCfg jobParams;
    private AddresseeDirectionDefaultCfg commonParams;

    protected override async Task<IEnumerable<IEvntFeed>>
    AgregateAsync(IEnumerable<IEvntFeed> events)
    {
        return events.Where(e => !string.IsNullOrEmpty(e.Flags) &&
            !string.IsNullOrEmpty(e.DueDocgroup)
            && e.Flags.Contains("ADD_ORG_ADDRESSEE") &&
            jobParams.Docgroups.Any(dg => e.DueDocgroup.StartsWith(dg)))
            .GroupBy(e => e.ObjectId)
            .Select(g => g.OrderBy(e => e.IsEvent).LastOrDefault())
            .OrderBy(e => e.IsEvent)
            .ToList();
    }

    protected override async Task EventProcessAsync(IEvntFeed evnt)
    {
        Logger.LogInformation("Обработка события № {0}, ПК {1}",
            evnt.IsEvent, evnt.ObjectId);
        //логика обработки
    }

    protected override async Task FillParamsAsync()
    {
        jobParams = await
            GetJobAppSettingsParamsAsync<AddresseeDirectionJobCfg>();
        commonParams = await
            GetDefaultAppSettingsParamsAsync<AddresseeDirectionDefaultCfg>();
        if (jobParams == null || commonParams == null)
            throw new NoDaemonConfigurationException();

        Delay.Minutes = (int)commonParams.Periodicity;
        await ValidateParamsAsync();
    }
}
```

При реализации класса, наследующего BaseDaemon:

Класс должен иметь переопределение методов метода WorkProcessAsync() – содержит основной код работы фоновой задачи.

Также могут быть переопределены методы Aborted(ThreadAbortException ex) – действия при прерывании работы ФЗ, Cancelled(OperationCanceledException ex) – действия при отмене ФЗ, Exception(Exception ex) – действия при возникновении ошибки, Finished() – действия при завершении работы ФЗ.

В теле метода WorkProcessAsync() должен содержаться вызов/вызовы метода CheckCancel() для осуществления остановки ФЗ в точках данной проверки при получении команды на остановку задачи (при остановке ФЗ из Управления ФЗ или при остановке пула). Если в WorkProcessAsync() присутствует бесконечный цикл while (true){ /* алгоритм работы ФЗ */ }, как для Автоматических ФЗ (DaemonKindType.Job), внутри цикла обязательно должен содержаться вызов CheckCancel(), иначе остановить ФЗ будет невозможно. Также, для циклов по Any(), где внутри цикла происходит загрузка значений в переменную, для которой вызывается метод Any(), необходимо добавлять вызов CheckCancel() перед загрузкой новых значений, например: while(events.Any()){ /*Обработка событий*/ CheckCancel(); events = GetNextEvent();}.

Для записи состояния ФЗ используются методы MarkInfoMessage(string message) - для сообщения ФЗ и MarkInfoError(string error)/MarkInfoError(Exception error) для записи ошибки ФЗ. Состояния отображаются в WEB-интерфейсе управления ФЗ.

Описание класса BaseDaemon, атрибутов и методов приведены в Приложение Б.

Пример класса:

```
// Пример класса
public class TestDispatcher : BaseDaemon
{
    protected override async Task WorkProcessAsync()
    {
        MarkInfoMessage("Выполнение ФЗ");
        //Основной код фоновой задачи
    }
}

Реализованные классы Фоновых задач необходимо зарегистрировать в классе
Startup, наследующем DaemonRegistrar (Eos.Platform.Daemon.Types), с указанием
описания ФЗ: группа ФЗ, название ФЗ, режим работы ФЗ, вид ФЗ, тип конфигурирования,
опциональные: типы конфигураций общих и экземплярных настроек, описания
обрабатываемых объектов и флагов.
// Регистрация классов ФЗ
public class Startup : DaemonRegistrar
{
    public override void ConfigureServices(IServiceCollection services)
    {
        base.ConfigureServices(services);
        services.AddAppSettingsTypeMappingServicePartWithDefaultLogic(new Hash
Set<Type>() { typeof(MailSenderCfg), typeof(AddresseeDirectionJobCfg),
            typeof(AddresseeDirectionDefaultCfg) });
    }

    public override IEnumerable<TypeInfo> GetTypeCollection()
    {
        var typeCollection = new TypeInfo[]
        {
            new TypeInfo(typeof(AddresseeDirectionDispatcher),
                "Примеры ФЗ",
                "Добавление организации-адресата",
                MultiInstanceType.Cooperative,
                DaemonKindType.Job,
                DaemonConfigType.UserConfig,
                typeof(AddresseeDirectionDefaultCfg), typeof(AddresseeDirectio
nJobCfg),
                "ADD_ORG_ADDRESSEE", "DOC_RC"),
            new TypeInfo(typeof(MailSenderDispatcher),
                "Примеры ФЗ",
                "Тестовая отправка Email",
                MultiInstanceType.Standby,
                DaemonKindType.Job,
                DaemonConfigType.UserConfig,
                null, typeof(MailSenderCfg)),
            new TypeInfo(typeof(TestDispatcher),
                "Примеры ФЗ",
                "Тестовая ФЗ на BaseDaemon",
                MultiInstanceType.Standby,
                DaemonKindType.Manual,
                DaemonConfigType.AutoConfig)
        };

        return typeCollection;
    }
}
```

}

Параметры конструктора класса описания ФЗ TypeInfo:

TypeInfo(...) – описание фоновой задачи

```
public TypeInfo(Type type, string group, string name, MultiInstanceType multiInstance,
DaemonKindType kind, DaemonConfigType configType, Type defaultConfigurationType =
null, Type jobConfigurationType = null, string flags = null, string kindObjects = null);
```

Для конструктора TypeInfo необходимо указать:

type - класс ФЗ;
group – группа, к которой принадлежит фоновая задача;
name – название ФЗ;
multiInstance – режим работы фоновой задачи, объект типа MultiInstanceType;
kind – вид ФЗ, объект типа DaemonKindType;
configType – способ конфигурирования ФЗ, объект типа DaemonConfigType.

Опционально:

defaultConfigurationType – тип класса конфигурации общих настроек (поля commonContext файла конфигурации);
jobConfigurationType – тип класса конфигурации экземплярных настроек (поля jobContext файла конфигурации);
flags – строка с указанием используемых флагов при отборе событий (справочная информация);
kindObjects – строка с указанием типов объектов при отборе событий (справочная информация).

```
public enum MultiInstanceType { Standby, Cooperative, Parallel }
```

Перечисление режимов работы фоновых задач:

Standby - вариант 'Singleton', ФЗ будет работать в единственном экземпляре на ВСЮ систему, назначается на один домен.

Cooperative - ФЗ может быть запущена сразу на нескольких доменах или в нескольких экземплярах на одном домене, для каждого экземпляра настраивается своя конфигурация, типа jobConfigurationType.

Parallel - режим, когда фоновая задача будет запущена на всех хостах в рамках назначенного(ых) домена(ов) одновременно.

```
public enum DaemonKindType { System, Job, Manual, Child }
```

Перечисление видов фоновых задач:

System – системная. ФЗ будет автоматически добавлена на системный домен.

Job – автоматическая. ФЗ выполняется с заданной периодичностью.

Manual – ручная. Выполнение ФЗ производится только после нажатия на кнопку «выполнить сейчас» в управлении фоновых задач (в ДЕЛО-WEB). Для повторного выполнения ФЗ необходимо снова нажать на кнопку выполнения.

Child – вспомогательная. ФЗ запускается родительской ФЗ, обрабатывает и завершается.

```
public enum DaemonTypeConfig { UserConfig, AutoConfig }
```

Перечисление способов конфигурирования фоновой задачи:

UserConfig - пользовательская конфигурация. Конфигурация задается с помощью файла Setup/*.json и настройки конфигурации в управлении фоновыми задачами.

AutoConfig – автоматическая конфигурация. Автоматически создается файл конфигурации фоновой задачи при запуске пула ФЗ.

Запуск экземпляров вспомогательных ФЗ

Фоновые задачи типа DaemonKindType.Child создаются и запускаются из родительской ФЗ вызовом метода RunChildDaemon<T>(...). Вспомогательные ФЗ

запускаются на том же хосте и домене, что и родительская ФЗ, отображаются в управлении фоновыми задачами как дочерние к родительской, действия над ними из web-интерфейса запрещены (запустить, остановить, выполнить сейчас, прервать выполнение).

Описание метода `RunChildDaemon<T>(...)` класса `BaseDaemon`:

`public async Task<T> RunChildDaemon<T>(Action<T> fillConfig = null, string displayName = "") where T : BaseDaemon`

Метод создает и запускает экземпляр вспомогательной (`DaemonKindType.Child`) фоновой задачи указанного типа `T`. Возвращает созданный экземпляр ФЗ.

Содержит необязательные параметры:

`fillConfig` - действие с экземпляром класса вспомогательной ФЗ по заполнению конфигурации. Например, `await RunChildDaemon<T>((instance) => { instance.Params = childParams; },)`, где `instance.Params` - поле вспомогательной ФЗ, а `childParams` - параметры, передаваемые из родительской ФЗ.

`displayName` - название экземпляра дочерней ФЗ, отображаемое в управлении фоновыми задачами.

6.3.3 Чтение конфигурации фоновой задачи

Каждая фоновая задача запускается с определенным набором параметров. Такими параметрами являются время создания, время последнего изменения, следующего запуска, сообщение ФЗ, ошибки и др. Они отображаются в таблице фоновых задач на странице управления ФЗ. Данные параметры обычно не используются напрямую при разработке ФЗ, время следующего запуска устанавливается при выполнении метода `GetNextDate`, сообщение устанавливается по методу `MarkInfoMessage`, ошибка по методу `MarkInfoError`.

Для классов, наследующих `BaseEventDispatcher` / `BaseDispatcher` (`Eos.Delo.Daemon.Dispatchers`), получение параметров ФЗ реализовано через методы `GetDefaultAppSettingsParamsAsync<T>()`, `GetJobAppSettingsParamsAsync<T>()`. Для классов, наследующих `BaseDaemon`, конфигурацию необходимо получать с помощью методов сервиса `IAppSettingsService`.

Для всех фоновых задач также доступен асинхронный метод получения идентификатора пользователя `CurrentInstance.VirtHost.GetCurrentUserId()`, указанного в настройках домена, по умолчанию `WFAGENT`. Данный параметр обычно используется при указании пользователя для контекста соединения с БД. Для классов, наследующих `BaseEventDispatcher` / `BaseDispatcher` (`Eos.Delo.Daemon.Dispatchers`) доступен метод `GetUserAsync()`, возвращающий пользователя.

Для фоновых задач, которые поддерживают работу в различных экземплярах, типа `MultiInstanceType.Cooperative`, указывается параметр `CurrentInstance.VirtualInstance.ConfigProfile`, содержащий идентификатор текущего выполняемого экземпляра.

Для получения пользовательской конфигурации (в случае, когда класс ФЗ зарегистрирован с `DaemonConfigType.UserConfig`) фоновой задачи используются методы `GetJobAppSettingsParamsAsync<T>()` – настройки экземпляра ФЗ (поля `jobContext` файла конфигурации) и `GetDefaultAppSettingsParamsAsync<T>()` – общие настройки ФЗ (поля `commonContext` файла конфигурации). В параметре `T` указывается **immutable** класс настроек, сохраняемых в таблице `APP_SETTINGS`. В проекте ФЗ необходимо описать **immutable** класс настроек, а также зарегистрировать в методе `ConfigureServices` класса `Startup`.

В классе настроек для параметра, содержащего список значений, например, множественный выбор из справочника, задается тип `IReadOnlyList<T>`, где `T` – тип элементов в списке. Для числовых значений задается тип `long`.

Работа с `AppSettings` описана в Приложении В.

Описание конфигурационного файла для ФЗ с пользовательской конфигурацией описано в пункте “4.1 Подготовка конфигурационного файла”.

Классы настроек криптографии и электронной почты содержатся в **Eos.Delo.Platform.Storage.Model**, с namespace Eos.Delo.Settings.Cryptography и в Eos.Platform.Notification.Email с namespace Eos.Platform.Settings.Email соответственно.

Пример класса конфигурации ФЗ:

```
// Класс конфигурации фоновой задачи
namespace Eos.Delo.Settings.TestDaemon
{
    public class DaemonCfg
    {
        public string Directory { get; init; }
        public bool CheckPlanDate { get; init; }
        public string EmailCfgInstanceId { get; init; }
        public string CryptographyCfgInstanceId { get; init; }
        public string Storage { get; init; }
    }
}
```

Получение параметров в коде ФЗ:

```
// Получение параметров фоновой задачи
using Eos.Delo.Daemon.Dispatchers;
using Eos.Delo.Platform.Storage.Extensions;
using Eos.Delo.Platform.Storage.Services.Caching;
using Eos.Delo.Settings.Cryptography;
using Eos.Delo.Settings.TestDaemon;
using Eos.Platform.AppSettings;
using Eos.Platform.Settings.Email;
using Eos.Platform.Settings.Abstractions;
using Microsoft.Extensions.Configuration;
using Microsoft.Extensions.DependencyInjection;
using System;
using System.IO;
using System.Threading.Tasks;

namespace Eos.Delo.TestDaemons.Daemons.Module
{
    public class TestDispatcher : BaseDispatcher
    {
        private DaemonCfg _cfg;
        private CommonCfg _emailCommonCfg;
        private SendCfg _emailSendCfg;
        private CryptographyCfg _cryptoCfg;
        private string _storagePath;

        protected override async Task FillParamsAsync()
        {
            MarkInfoMessage("Получение параметров ФЗ");
            await InContextAsync(async (context, provider) => {
                var appSettingsSrv =
provider.GetRequiredService<IAppSettingsService>();

                _cfg = await appSettingsSrv.GetAsync<DaemonCfg>();
                //Общие настройки email
                _emailCommonCfg = await
appSettingsSrv.GetAsync<CommonCfg>(_cfg.EmailCfgInstanceId);
                //Настройки отправки email
                _emailSendCfg = await
appSettingsSrv.GetAsync<SendCfg>(_cfg.EmailCfgInstanceId);
                //Настройки криптографии
                _cryptoCfg = await
appSettingsSrv.GetAsync<CryptographyCfg>(_cfg.CryptographyCfgInstanceId);

                var storageCfg = await
appSettingsSrv.GetAsync<SmbCfg>(_cfg.Storage);
```

```

        //определение пути к хранилищу зависит от ОС
        if (OperatingSystem.IsLinux())
        {
            var baseVarDirPath =
provider.GetRequiredService<IConfiguration>().GetValue<string>("Eos:BaseVarDir
Path");
            _storagePath = Path.Combine(baseVarDirPath,
storageCfg.MountName);
        }
        else
            _storagePath = storageCfg.StoragePath;
    });
}

protected override async Task WorkAsync()
{
    MarkInfoMessage(TestLocalization.StartWorkAsync);
    //Обработка объектов фоновой задачей
}
}
}

```

Для загрузки конфигурации ФЗ в классе-наследнике BaseDaemon используется сервис IAppSettingsService. Также для ДЕЛЮ API 2009.

Пример получения конфигурации:

```

using IServiceScope serviceScope =
DaemonHost.ServiceProvider.GetRequiredService<IServiceScopeFactory>().CreateSc
ope();
IServiceProvider serviceProvider = serviceScope.ServiceProvider;
var appSettingsService =
serviceProvider.GetRequiredService<IAppSettingsService>();
var commonParams =
appSettingsService.GetAsync<AddresseeDirectionCommonCfg>().Result;
var jobParams =
appSettingsService.GetAsync<AddresseeDirectionJobCfg>(CurrentInstance.VirtualI
nstance.ConfigProfile).Result;

```

Для регистрации класса для работы через API с использованием стандартного обработчика MutationLogic (проверка права системного технолога и права Параметры системы) используйте метод расширения в Eos.Delo.Platform.Storage.Extensions в ConfigureServices:

```

// Регистрация класса для работы через API
public class Startup : DaemonRegistrator
{
    public void ConfigureServices(IServiceCollection services)
    {
        base.ConfigureServices(services);
        services.AddAppSettingsTypeMappingServicePartWithDefaultLogic(new Hash
Set<Type>() { typeof(DaemonCfg) });
    }
    public override IEnumerable<TypeInfo> GetTypeCollection()
    {
        var typeCollection = new TypeInfo[]
        {
            new TypeInfo(typeof(TestDispatcher),
                "Примеры ФЗ",
                "Тестовая ФЗ",
                MultiInstanceType.Standby,
                DaemonKindType.Manual,
                DaemonConfigType.UserConfig,
                typeof(DaemonCfg)
            );
        };
    }
}

```

```

        return typeCollection;
    }
}

```

Пример регистрации класса см. в дистрибутиве системы в каталоге «Sdk» (проект Eos.Delo.Samples.Daemons).

6.3.4 Соединение с базой данных

Для соединения с базой данных, а также работы с другими сервисами, необходимо получить ServiceProvider через создание ServiceScope, который необходимо использовать через using или закрывать явным Dispose(). ServiceScope, ServiceProvider, IDataContext нельзя хранить в свойствах/полях долгоживущих классов, удерживая соединение с БД.

Для работы с ServiceProvider, IDataContext через классы, наследующие базовые диспетчеры, используется метод **InContextAsync(Func<IDataContext, IServiceProvider, Task> action)** или **InContextAsync(Func<IDataContext, Task> action)**, если не требуется доступ к ServiceProvider.

Для того, чтобы запустить запрос через Middleware, необходимо получить новый экземпляр MiddlewareContextBase. Для этого нужен ClaimsPrincipal, который можно сделать на основании идентификатора пользователя CurrentInstance.VirtHost.GetCurrentUserId(). Также необходимо получить текущего пользователя через UserCachingService, для базовых диспетчеров Дело получение пользователя равносильно вызову метода GetUserAsync(), и сформировать новый DeloContext.

Ниже приведен пример кода соединения с базой данных:

```

// Соединение с базой данных
public class TestDispatcher : BaseDispatcher
{
    protected override async Task FillParamsAsync()
    {
        //Получение параметров
    }

    protected override async Task WorkAsync()
    {
        await InContextAsync(async (db, serviceProvider) =>
        {
            var user = await GetUserAsync();
            //Запуск обычного запроса
            var docRc = await db.DocRcs.Where(p => p.IsnDoc == 1)
                .AsNoTracking()
                .FirstOrDefaultAsync();

            //Запуск запроса через Middleware
            var queryContext = MiddlewareHelper.CreateBaseContext(db, user);

            var docRc2 = await db.DocRcs.Where(p => p.IsnDoc == 1)
                .AsNoTracking()
                .FirstOrDefaultAsync(serviceProvider, new QueryContext<IDocRc>
(queryContext));
        });
    }
}

```

6.3.5 Поиск объектов

Поиск объектов можно производить через IDataContext, либо используя методы расширения, для прогона запроса через конвейер middleware.

Пример запроса поиска объектов:

```
// Запрос поиска объектов
await InContextAsync(async (db, serviceProvider) =>
{
    var userId = await CurrentInstance.VirtHost.GetCurrentUserId();
    var userService =
serviceProvider.GetRequiredService<UserCachingService>();
    var user = await userService.GetUserAsync(userId);
    //Простой запрос
    var docRc = await db.DocRcs
        .Where(p => p.IsnDoc == 1 && EF.Functions.Like("тест", p.Annotat
+ "%"))
        .AsNoTracking()
        .FirstOrDefaultAsync();

    //Запрос с заходом через middleware
    var queryContext = MiddlewareHelper.CreateBaseContext(db,
user, true, null, extProtOperationType: ExtProtOperationType.Query);

    var docRc2 = await db.DocRcs.Where(p => p.IsnDoc == 1)
        .AsNoTracking()
        .FirstOrDefaultAsync(serviceProvider, new QueryContext<IDocRc>(queryCo
ntext));
});
```

6.3.6 Поиск ссылки на сущность системы

Для получения ссылки на сущность системы можно использовать методы Include и ThenInclude.

Пример поиска ссылки на сущность системы:

```
// Поиск ссылки на сущность системы
var department = await db.Departments.Where(p => p.IsnNode == 3)
    .Include(p => p.ParentNode)
    .ThenInclude(p => p.ParentNode)
    .AsNoTracking()
    .FirstOrDefaultAsync(_serviceProvider, new QueryContext<IDepartment>(query
Context));

var dep3 = department.ParentNode.ParentNode;
```

* примечание: не рекомендуется строить запросы с большим количеством Include, если Include возвращают IEnumerable (например, для Department это ChildNodes), чтобы избежать [Cartesian Explosion](#).

6.3.7 Загрузка объекта по идентификатор

Часто при разработке фоновых задач необходимо получить (загрузить) данные единственного экземпляра объекта по известному системному коду. Для загрузки можно использовать методы EntityHelpersService.

Примеры кода можно посмотреть в дистрибутиве системы в каталоге «Sdk» (проект Eos.Delo.Samples.Daemons).

6.3.8 Модификация данных объекта

Как правило, фоновые задачи создаются для выполнения различных модификаций объектов системы. Для модификаций объектов используются стандартные методы Entity Framework, либо методы расширения, чтобы запустить конвейер middleware для сущностей,

Пример добавления РК документа:

```
// Добавление РК документа
var docRc = _dataContext.CreateDocRc();
docRc.IsnDoc =
_dataContextHelper.GetSeqAsync(SeqConstants.SEQ_DOCPRJ_RC).Result;
docRc.DocDate = DateTime.UtcNow.Date;
docRc.IsnDocvid = 0;
```

```

docRc.FreeNum = Guid.NewGuid().ToString("N");
docRc.Consists = "1";
docRc.Securlevel = 1;
docRc.DueDocgroup = "0.2TZ.2U7.";
docRc.Annotat = "Содержание РК";
docRc.AdressFlag = 3;
_dataContext.Add(docRc);
var resSave = _dataContext.SaveChangesAsync().Result;

```

Пример добавления РК документа, с запуском конвейера middleware:

```

// Добавление РК документа с запуском конвейера middleware
var docRc = _dataContext.CreateDocRc();
docRc.DocDate = DateTime.UtcNow;
docRc.FreeNum = Guid.NewGuid().ToString("N");
docRc.Consists = "1";
docRc.Annotat = "Содержание РК";
_dataContext.Add(docRc);

var user = await GetUserAsync();
var mutationContext = MiddlewareHelper.CreateMutationContext(_dataContext,
user, true, null,
    ExtProtOperationType.Mutation, Token);
var resSave = await _dataContext.SaveChangesAsync(_serviceProvider,
mutationContext);

```

Пример добавления подразделения и ДЛ внутри него:

```

// Добавление подразделения и ДЛ внутри него
var department = _dataContext.CreateDepartment();
var parentNode = _dataContext.CreateDepartment();

department.ClassifName = "Тест";
department.ParentNode = parentNode;

parentNode.IsnHighNode = 1;
parentNode.ClassifName = "Тест dep";

_dataContext.Add(department);
_dataContext.Add(parentNode);

var user = await GetUserAsync();
var mutationContext = MiddlewareHelper.CreateMutationContext(_dataContext,
user, true, null,
    ExtProtOperationType.Mutation, Token);

var resSave = await _dataContext.SaveChangesAsync(_serviceProvider,
mutationContext);

```

Пример изменения и удаления сущности:

```

// Изменение и удаление сущности
var docRc = await _dataContext.DocRcs.Where(b => b.IsnDoc == 1)
    .AsNoTracking()
    .FirstOrDefaultAsync();

var docRc2 = await _dataContext.DocRcs.Where(b => b.IsnDoc == 2)
    .AsNoTracking()
    .FirstOrDefaultAsync();

_dataContext.EnsureTracked(docRc);
docRc.Annotat = "Тест1";

_dataContext.Remove(docRc2);

var user = await GetUserAsync();

var mutationContext = MiddlewareHelper.CreateMutationContext(_dataContext,

```

```

user, true, null,
    ExtProtOperationType.Mutation, Token);

var resSave = await _dataContext.SaveChangesAsync(_serviceProvider,
mutationContext);

```

6.3.9 Работа с email сообщениями

Отправка email по каналу IEmailChannel:

Для отправки email по каналу IEmailChannel необходимо подключить к проекту NuGet пакет Eos.Platform.Notification.Email и использовать метод bool Send(string profileName, IAppSettingsService appSettingsService, ISecretStore secretStore, string addressee, string messageText, ILogger logger) возвращающий true/false в зависимости от успешности отправки сообщения.

Метод принимает параметры:

string profileName - профиль электронной почты (значение поля INSTANCE таблицы APP_SETTINGS)

IAppSettingsService appSettingsService - сервис настроек AppSettings

ISecretStore secretStore - сервис секретов (для паролей)

string addressee - email адрес получателя сообщения

string messageText - email сообщение

ILogger logger - логгер

Пример использования метода Send:

```

public void SendEmail(string emailCfgInstanceId, string emailMessage, string
addressee, IServiceProvider services, ILogger logger)
{
    var channel = services.GetRequiredService<IEmailChannel>();

    var appSettingsService =
services.GetRequiredService<IAppSettingsService>();
    var secretStore = services.GetRequiredService<ISecretStore>();

    var success = channel.Send(emailCfgInstanceId, appSettingsService,
secretStore, addressee, emailMessage, logger);
}

```

Работа с email сообщениями с помощью MailManHelper, библиотека Eos.Delo.Utils:

Методы используют файловую систему для хранения сообщений электронной почты в стандартном формате “.eml”. При ошибках выполнения отправки и получения email сообщений, а также указания неизвестного типа авторизации для SMTP сервера, вызываются исключения.

Отправка сообщения

Для отправки сообщений через SMTP сервер используется метод SendEmail. При создании экземпляра класса MailManHelper необходимо указать настройки для SMTP сервера типа SmtпSettings. Для метода SendEmail указываются путь к электронному сообщению, необходимость отправки в зашифрованном виде и отправки с цифровой подписью.

Пример отправки сообщения:

```

// Отправка email сообщения
var smtpSetting = new SmtпSettings()
{
    Port = 465,
    AuthMethod = "LOGIN",
    Host = "smtp.mail.ru",
    Username = "login@mail.ru",
    Password = "password"
};
var mailMan = new MailManHelper(smtpSetting);

```

```
try
{
    mailMan.SendEmail(pathEml, false, false);
}
catch { ... }
```

Получение сообщения

Для получения (выгрузки) сообщения с сервера POP3 используется метод `GetEmail`. При создании экземпляра класса `MailManHelper` необходимо указать настройки для POP3 сервера типа `PopSettings`. Для метода `GetEmail` указываются путь к директории для хранения сообщений и поле типа `string`, для записи пути к выгруженному сообщению. Полученные сообщения удаляются с сервера.

Пример получения сообщения:

```
// Получение email сообщения
var popSetting = new PopSettings()
{
    Port = 995,
    Host = "pop.mail.ru",
    Login = "login@mail.ru",
    Password = "password"
};
var mailMan = new MailManHelper(popSetting);
try
{
    mailMan.GetEmail(pathDirectory, out string emlReceivedPath);
}
catch { ... }
```

Также работать с email сообщениями можно стандартными средствами .NET Core:

```
// Работа с email сообщениями
var message = new MailMessage();
message.IsBodyHtml = false;
message.From = new MailAddress(MailFrom);
message.To.Add(EMail);

message.Subject = "тема";
message.Body = "текст";

message.BodyEncoding = Encoding.UTF8;

SmtpClient smtp = new SmtpClient(SMTPAddress, SMTPPort);
smtp.Credentials = new NetworkCredential(Login, Pass, Domain);
smtp.Send(message);
```

6.3.10 Использование спецхранилища значений (CUSTOM_STORAGE)

Для хранения дополнительной информации, для которой не нашлось места в структурах стандартных объектов системы «ДЕЛО», в базе данных имеется таблица `CUSTOM_STORAGE`.

В `CUSTOM_STORAGE` можно хранить данные, ассоциированные с любым объектом и даже с дочерней сущностью объектов, без типизации хранимых данных – все данные хранятся в виде строк.

Хранение информации ФЗ в `CUSTOM_STORAGE` позволяет организовать взаимодействие между ФЗ, сохранять информацию о последнем обработанном событии и др.

Структура таблицы `CUSTOM_STORAGE`:

- `VALUE_ID` – идентификатор хранимого значения
- `VALUE` – хранимое значение
- `OWNER_KIND` – тип объекта владельца
- `OWNER_ID` – идентификатор объекта владельца

- ORDERNUM – порядковый номер записи.

С одним идентификатором VALUE_ID может быть связан список значений. Значение ORDERNUM указывает на порядковый номер записи в этом списке.

Для доступа к CUSTOM_STORAGE, загрузки, сохранения и удаления записей, используется интерфейс ICustomStorage.

```
// Доступ к CUSTOM_STORAGE
```

```
var customStorage = await dataContext.CustomStorages.FirstOrDefaultAsync();
```

6.3.11 Работа с событиями

При работе в системе «ДЕЛО» происходят различные события, сведения о которых сохраняются в базе данных. При работе триггеров БД или ФЗ «Планировщик задач Базы Данных» происходит заполнение таблицы EVNT_FEED всеми происходящими в системе событиями. Записи таблицы содержат ограниченный набор данных о событии:

- Тип события: создание или модификация (в т.ч. удаление) объекта
- Идентификатор объекта события
- Группа документов объекта (для РК, РКПД, поручений)
- Флаги
- Метка времени
- Идентификатор события

Подробно механизм событий описан в Приложение Л.

Для работы ФЗ с событиями, выполняются запросы к БД с критериями поиска необходимых событий, которые будут в дальнейшем обрабатываться. Чтобы сохранять информацию об уже обработанных событиях в ФЗ, сохраняется запись в CUSTOM_STORAGE – ватермарка, со значением номера (ISL_EVENT) последнего события, прошедшего обработку. Поиск событий для последующей обработки осуществляется, начиная с данной ватермарки. Если предполагается, что ФЗ будет иметь несколько экземпляров работы или запущена с нескольких хостов, подключенных к одной БД, для каждого экземпляра нужна своя ватермарка (реализовано в Eos.Delo.Daemon.Dispatchers.BaseDispatcher, ватермарка содержит идентификатор экземпляра ФЗ).

Пример сохранения значения ватермарки:

```
// Сохранение значения ватермарки
var customStorage = _dataContext.CreateCustomStorage();
customStorage.ValueId = WATERMARK_ID;
customStorage.Value = value.ToString();
customStorage.OwnerKind = 0;
customStorage.OwnerId = null;
customStorage.OrderNum = 1;
_dataContext.Add(customStorage);
var resSave = await _dataContext.SaveChangesAsync();
```

Пример получения значения ватермарки:

```
// Получение значения ватермарки
var customStorage = await _dataContext.CustomStorages.Where(p => p.ValueId ==
WATERMARK_ID).AsNoTracking().SingleOrDefaultAsync();
var value = customStorage.Value;
```

Пример работы с событиями, для наследника BaseDaemon (в BaseEventDispatcher работа с событиями уже реализована):

```
// Работа с событиями для наследника BaseDaemon
private readonly string WATERMARK_ID = typeof(TestDispatcher).FullName
+ ".Watermark";
private const int SIZE = 500;

protected override async Task WorkProcessAsync()
```

```

{
    Delay.Seconds = 30;
    MarkInfoMessage("Запуск Ф3");

    while (true)
    {
        CheckCancel();
        try
        {
            var watermark = await LoadWatermarkAsync();
            var events = await GetNextEventsAsync(watermark);

            if (events.Any())
            {
                MarkInfoMessage("Обработка событий");

                var aggregate = events.Where(e =>
e.Flags.Contains("I_REF_SEND"))
                    .OrderBy(e => e.IsEvent)
                    .ToList();

                //Обработка событий

                var lastEvent = events.LastOrDefault();
                watermark = lastEvent == null ? watermark :
(long)lastEvent.IsEvent + 1;
                await SaveWatermarkAsync(watermark);
            }
        }
        catch (Exception ex)
        {
            MarkInfoError(ex);
        }
        MarkInfoMessage("Ожидание событий");
        await WaitAsync();
    }
}

private async Task<long> LoadWatermarkAsync()
{
    using var scope =
DaemonHost.ServiceProvider.GetRequiredService<IServiceScopeFactory>().CreateSc
ope();
    var db = scope.ServiceProvider.GetRequiredService<IDataContext>();
    var watermark = (await
db.CustomStorages.AsNoTracking().SingleOrDefaultAsync(cs => cs.ValueId ==
WATERMARK_ID)).Value ?? "0";
    return long.Parse(watermark);
}

private async Task SaveWatermarkAsync(long value)
{
    using var scope =
DaemonHost.ServiceProvider.GetRequiredService<IServiceScopeFactory>().CreateSc
ope();
    var db = scope.ServiceProvider.GetRequiredService<IDataContext>();
    var customStorage = await db.CustomStorages.Where(p => p.ValueId ==
WATERMARK_ID).AsNoTracking().FirstOrDefaultAsync();

    if (customStorage == null)
    {
        customStorage = db.CreateCustomStorage();
        customStorage.ValueId = WATERMARK_ID;
    }
}

```

```

        customStorage.Value = value.ToString();
        customStorage.OwnerKind = 0;
        customStorage.OwnerId = null;
        customStorage.Ordernum = 1;
        db.Add(customStorage);
    }
    else
    {
        db.EnsureTracked(customStorage);
        customStorage.Value = value.ToString();
    }

    await db.SaveChangesAsync();
}

private async Task<List<IEvntFeed>> GetNextEventsAsync(long watermark)
{
    using var scope =
        DaemonHost.ServiceProvider.GetRequiredService<IServiceScopeFactory>().CreateScope();
    var db = scope.ServiceProvider.GetRequiredService<IDataContext>();
    return await db.EvntFeeds.Where(p => p.IsEvent >= watermark &&
        p.KindObject == 1 && p.KindEvent == FeedEventKind.Creation)
        .AsNoTracking()
        .OrderBy(p => p.IsEvent)
        .Take(SIZE)
        .ToListAsync();
}

```

WorkProcessAsync – метод работы ФЗ. Данный пример для автоматической ФЗ, поэтому указывается интервал запуска `Delay.Seconds`, ожидание `WaitAsync` с вычислением времени следующего запуска. При наличии событий, выполняется отбор событий в aggregate, с флагом “I_REF_SEND”, т.е. с добавлением адресата. Таким образом, в aggregate содержатся события с добавлением адресатов при создании РК.

LoadWatermarkAsync – метод загрузки значения ватермарки.

SaveWatermarkAsync – метод сохранения значения ватермарки.

GetNextEventsAsync – метод получение событий с созданием РК (`KIND_EVENT = 1`, `KIND_OBJECT = 1`), количество событий указано в `SIZE`.

Данный пример учитывает работу только одного экземпляра ФЗ. Если предполагается работа нескольких экземпляров, нужно использовать свою ватерматрку для каждого экземпляра (`BaseDaemon`).

Примеры фоновых задач можно посмотреть в дистрибутиве системы в каталоге «Sdk» (проект `Eos.Delo.Samples.Daemons`).

6.4 Подготовка фоновой задачи к выполнению

6.4.1 Подготовка конфигурационного файла

Для фоновой задачи с пользовательским типом конфигурации `UserConfig`, в проекте необходимо создать конфигурационный файл для настройки.

Конфигурационный файл для настройки ФЗ представляет собой файл с расширением “json”, расположенный в папке «Setup» проекта. Имя такого файла формируется как имя класса, наследующего `BaseDispatcher` / `BaseEventDispatcher` / `BaseDaemon` + “json”. Например, если класс ФЗ – “`TestDispatcher`”, тогда имя конфигурационного файла «`TestDispatcher.json`».

В свойствах файла необходимо указать значение свойства «Действие при сборке» - «Внедренный ресурс» («`Build Action`» - «`Embedded resource`») и «Копировать в выходной каталог» - «Не копировать» («`Copy to Output Directory`» - «`Do not copy`»).

При добавлении конфигурационного файла, в файле проекта появится запись:

```
<ItemGroup>
```

```
<EmbeddedResource Include="Setup\TestDispatcher.json">
  <CopyToOutputDirectory>Never</CopyToOutputDirectory>
</EmbeddedResource>
</ItemGroup>
```

Конфигурационный файл имеет следующий шаблон:

```
{
  "commonContext": {
    "fields": [],
    "rows": [],
    "defaultValues": {}
  },
  "jobContext": {
    "fields": [],
    "rows": [],
    "defaultValues": {}
  }
}
```

Объекты:

- "commonContext" – содержит описание полей настроек, значения параметров которых общие для всех экземпляров ФЗ
- "jobContext" – содержит описание полей настроек, значения параметров которых различны для экземпляров ФЗ

Объекты "commonContext" и "jobContext" оба могут содержать описания полей настройки, либо только один (как в шаблоне выше), в зависимости от конфигурации ФЗ. Данные объекты представляют контекст описания полей настройки с помощью объектов "fields" и "rows", также может быть описан или отсутствовать "defaultValues".

- "fields" – список полей настроек с их описанием
- "rows" – размещение полей на строках
- "defaultValues" – значения полей по умолчанию

Типы полей:

- FieldAutoComplete – поле выбора из справочников
- FieldCheckbox – флаг (чекбокс)
- FieldEnum – выбор из списка заданных значений
- FieldDateTime – выбор даты из календаря
- FieldInteger – числовое поле
- FieldText – текстовое поле
- FieldTimePicker – поле выбора времени
- FieldPassword – текстовое поле для пароля
- FieldProfile – выбор профиля настроек
- FieldStorage – выбор хранилища
- FieldButton – кнопка

Базовые настройки полей:

Обязательные:

- "name" - имя поля в форме (строка)
- "type" – тип поля (строка)

Необязательные:

- "label" - отображаемое наименование поля (строка)
- "required" - обязательность поля (true / false)
- "requiredMessage" - сообщение об обязательности поля (строка)
- "disabled" – недоступность поля (true / false)
- "placeholder" - html атрибут placeholder (строка)
- "hidden" – скрытость поля (true / false)

Примеры полей по типам:

6.4.1.1 FieldAutoComplete – поле выбора из справочника

```
{
    "label": "Организация",
    "name": "Organization",
    "required": true,
    "type": "FieldAutoComplete",
    "multiple": false,
    "chooseClassif":
    {
        "classif": "ORGANIZ_CL",
        "multi": false,
        "returnDue": true
    }
}
```

Данный пример описывает поле с отображаемым названием "Организация" (label), обязательное для заполнения ("required": true), с выбором единственного значения ("multiple": false и "multi": false).

Объект "chooseClassif" описывает параметры выбора из справочника:

"classif" – поле типа справочника, указывается в соответствии с названиями таблиц для объектов, входящих в справочники. Может принимать следующие значения:

- "LINK_CL" – связи
- "DEPARTMENT" – подразделения
- "CARDINDEX" – картотеки
- "DELIVERY_CL" – виды доставки
- "DOCGROUP_CL" – группы документов
- "RUBRIC_CL" – рубрикатор
- "SECURITY_CL" – грифы доступа
- "LINK_CL" – типы связей
- "ORGANIZ_CL" – организации
- "USER_CL" – пользователи
- и другие...

"multi" – флаг возможности выбора нескольких значений (true / false);

"returnDue" – true: возвращает due выбранных объектов справочника; false: возвращает isn объекта.

Также в "chooseClassif" могут быть указаны:

- "nodes" - флаг возможности выбора вершин (true / false),
- "leafs" - флаг возможности выбора листьев (true / false)

При сохранении значений конфигурации, при возможности выбора одного значения ("multiple": false и "multi": false) значение будет записано как строковое значение равное due. Для справочников с isn объектов (например, связи) в будет записано числовое значение isn. В случае множественного выбора ("multiple": true и "multi": true), будет записан массив таких значений.

Если необходимо указать значение по умолчанию, то это значение указывается в "defaultValues":

```
"defaultValues": {"Organization": "0.393.2EZ0H."}
```

При написании значений по умолчанию необходимо учитывать, является данное поле для значений множественного выбора или нет. Для множественного выбора, значения указываются внутри массива:

```
"defaultValues": {"Organization": ["0.393.2EZ0H.", "0.391."]}
```

6.4.1.2 FieldCheckbox – флаг (чекбокс)

```
{
    "name": "RegistrationOnly",
    "type": "FieldCheckbox",
```

```

        "description": "Отправлять только при регистрации РК"
    }

```

"description" – поле описание чекбокса. Это текстовая строка, отображаемая рядом с чекбоксом.

Флаг (чекбокс) принимает значение true/false.

6.4.1.3 FieldEnum – выбор из списка заданных значений

```

{
    "label": "Тип",
    "name": "Type",
    "type": "FieldEnum",
    "required": true,
    "options": [{ "value": 1, "label": "Оригинал" },
                 { "value": 2, "label": "Любой экземпляр" },
                 { "value": 3, "label": "Копия" } ]
}

```

"options" – массив значений для выбора.

При необходимости указать значения по умолчанию, значения указываются в "defaultValues":

"defaultValues": {"Type": 2}

Если выбор значения должен влиять на заполняемость других полей, необходимо описать объект "onChange":

```

{
    "label": "Метод авторизации",
    "name": "AuthorizationMethod",
    "type": "FieldEnum",
    "required": true,
    "options": [{ "value": "NONE", "label": "Авторизация не требуется" },
                 { "value": "LOGIN", "label": "Авторизация логин/пароль" },
                 { "value": "NTLM", "label": "Авторизация Windows" } ],
    "onChange": [{ "action": "disableField", "field": "SmtptUsername",
                   "value": "NONE" },
                  { "action": "disableField", "field": "SmtptPassword", "value":
"NONE" },
                  { "action": "enableField", "field": "SmtptUsername", "value":
"LOGIN" },
                  { "action": "enableField", "field": "SmtptPassword", "value":
"LOGIN" },
                  { "action": "enableField", "field": "SmtptUsername", "value":
"NTLM"},
                  { "action": "enableField", "field": "SmtptPassword", "value":
"NTLM" } ]
}

```

При наличии блока "onChange" обязательно указание значения по умолчанию в "defaultValues".

В "onChange" представлен массив с описанием действий при изменении выбора:

- "action" – действие (disableField / enableField, hideField / showField, setRequiredField / unsetRequiredField),
- "field" – поле, над которым производится действие,
- "value" – выбранное значение.

В данном примере, при значении "Авторизация не требуется" -"NONE", поля с name "SmtptUsername" и "SmtptPassword" будут не активны, в остальных случаях – активны.

Если вместо "disableField" указать "hideField", а вместо "enableField" - "showField", то поля имени и пароля будут скрыты при значении "NONE" и видимы в других значениях.

Разница использования "disableField"-"enableField" или "hideField"-"showField" состоит не только в визуальном отображении, но также и при сохранении конфигурации. В первом случае, конфигурация будет содержать пустые значения полей "SmtпUsername" и "SmtпPassword" или заполненные при выборе другого значения переключателя. Во втором случае, в сохраненной конфигурации будут отсутствовать значения полей "SmtпUsername" и "SmtпPassword".

Также, "action" может содержать действие "setRequiredField" или "unsetRequiredField" устанавливающее или снимающее обязательность заполнения с видимого поля, указанного в "field", при выбранном "value".

Для множественного выбора указывается "enumType": "multiple"

```
{
    "label": "Типы",
    "name": "Types",
    "type": "FieldEnum",
    "enumType": "multiple",
    "options": [{ "value": 1, "label": "Тип 1" },
                { "value": 2, "label": "Тип 2" },
                { "value": 3, "label": "Тип 3" } ]
}
```

В таком случае, в конфигурацию сохраняется массив выбранных значений.

В значениях по умолчанию также указывается массив значений value:

"defaultValues": {"Types": [1]}

6.4.1.4 FieldDateTime – выбор даты из календаря

```
{
    "label": "Начиная с даты",
    "name": "StartDate",
    "type": "FieldDateTime",
    "minDate": "2021",
    "maxDate": "05.25.22"
}
```

- "minDate" – минимальная дата выбора из календаря,
- "maxDate" – максимальная дата.

Эти поля принимают значения как строку в формате мм.дд.гггг / мм.дд.гг / гггг.

Данные настройки поля необязательны.

Значение по умолчанию указывается в формате строки гггг.дд.мм:

"defaultValues": {"StartDate": "2022.16.04"}

Значение сохраняется в формате строки "yyyy-DD-MMTTHH:mm:ss".

6.4.1.5 FieldInteger – числовое поле

```
{
    "label": "Интервал запуска",
    "name": "Delay",
    "type": "FieldInteger",
    "required": true
}
```

- "min" - минимальное значение,
- "max" - максимальное значение.

6.4.1.6 FieldText – текстовое поле

```
{
    "label": "Название папки",
    "name": "Folder",
    "type": "FieldText",
    "required": true
}
```

- "minLength" - минимальная длина (целое число),

– "maxLength" - максимальная длина (целое число)

6.4.1.7 FieldTimePicker – поле выбора времени

```
{
    "label": "Время",
    "name": "Time",
    "required": true,
    "type": "FieldTimePicker"
}
```

6.4.1.8 FieldPassword – текстовое поле для пароля

```
{
    "label": "Пароль",
    "name": "Password",
    "type": "FieldPassword",
    "required": true
}
```

В данном поле текст скрыт.

6.4.1.9 FieldProfile – выбор профиля настроек

```
{
    "label": "Профиль настроек",
    "name": "ConfigProfile",
    "type": "FieldProfile",
    "required": true,
    "selectProfile": {
        "namespace": "Eos.Delo.Settings.Box",
        "typename": "DaemonCfg",
        "propertyProfileName": "ProfileName"
    }
}
```

Контроль используется для выбора профиля из app_settings по заданным namespace, typeName и propertyProfileName, в котором указывается имя поля (property) настройки отвечающее за название профиля. Данные параметры указываются в описании "selectProfile".

Для выбора профиля электронной почты:

```
{
    "label": "Профиль эл. почты",
    "name": "EmailProfile",
    "type": "FieldProfile",
    "required": true,
    "selectProfile": {
        "namespace": "Eos.Platform.Settings.Email",
        "typename": "CommonCfg",
        "propertyProfileName": "ProfileName"
    }
}
```

Для выбора профиля криптографии:

```
{
    "label": "Профиль криптографии",
    "name": "CryptoProfile",
    "type": "FieldProfile",
    "required": true,
    "selectProfile": {
        "namespace": "Eos.Delo.Settings.Cryptography",
        "typename": "CryptographyCfg",
        "propertyProfileName": "ProfileName"
    }
}
```



```
}
```

Необходимо использовать выбор единственного значения "multiple": false (установлен по умолчанию, можно не указывать). Т.к. один экземпляр работы фоновой задачи должен отвечать за один профиль настройки.

6.4.1.10 FieldStorage – выбор хранилища

```
{
    "label": "Хранилище",
    "name": "Storage",
    "type": "FieldProfile",
    "required": true,
    "selectProfile": {
        "namespace": Eos.Platform.Settings.Abstractions",
        "typename": "SmbCfg",
        "propertyProfileName": "DisplayName"
    }
}
```

6.4.1.11 FieldButton – кнопка

```
{
    "type": "FieldButton",
    "name": "SendButton",
    "onClick": {
        "TestSendEmail": {
            "EmailProfile": "EmailProfile",
            "ToAddress": "To"
        }
    }
}
```

Поле с кнопкой поддерживает метод, позволяющий сделать тестовую отправку сообщения.

Если требуется проверить отправку email сообщения, необходимо в "onClick" указать, что требуется выполнить метод "TestSendEmail", со следующими параметрами "EmailProfile": "EmailProfile", "ToAddress": "To". Параметры "EmailProfile", "ToAddress" фиксированные, в них указывается имя поля, из которого нужно получить соответствующее значение. Поле, указанное для "EmailProfile" должно быть типа FieldProfile с выбором профиля электронной почты. Поле для "ToAddress" должно содержать email адрес получателя. Пример полной конфигурации с данным методом кнопки указан ниже. Также, данная конфигурация содержится в примере ФЗ в дистрибутиве системы в каталоге «Sdk» (проект Eos.Delo.Samples.Daemons).

Пример конфигурации с кнопкой тестовой отправки email:

```
{
  "jobContext": {
    "fields": [
      {
        "label": "Профиль эл. почты",
        "name": "EmailProfile",
        "type": "FieldProfile",
        "required": true,
        "selectProfile": {
          "namespace": "Eos.Platform.Settings.Email",
          "typename": "CommonCfg",
          "propertyProfileName": "ProfileName"
        }
      },
      {
        "label": "Почтовый адрес администратора",
```

```

        "name": "Admin",
        "type": "FieldText",
        "required": true,
        "requiredMessage": "Обязательное поле"
    },
    {
        "type": "FieldButton",
        "name": "TestSendButton",
        "onClick": {
            "TestSendEmail": {
                "EmailProfile": "EmailProfile",
                "ToAddress": "Admin"
            }
        }
    }
],
"rows": [
    {
        "cells": [
            {
                "fields": [ "EmailProfile" ],
                "width": 24
            }
        ]
    },
    {
        "cells": [
            {
                "fields": [ "Admin" ],
                "width": 24
            }
        ]
    },
    {
        "cells": [
            {
                "fields": [ "TestSendButton" ],
                "width": 24
            }
        ]
    }
],
"defaultValues": {}
}

```

Пример конфигурационного файла:

```

{
    "commonContext": {
        "fields": [
            {
                "label": "Интервал запуска",
                "name": "Periodicity",
                "type": "FieldInteger",
                "required": true,
                "max": 99,
                "min": 1
            }
        ]
    },
    "rows": [
        {
            "cells": [
                {

```

```

        "fields": [ "Periodicity" ],
        "width": 12
    }
]
},
"defaultValues": { "Periodicity": 1 }
},
"jobContext": {
    "fields": [
        {
            "label": "Организация",
            "name": "Organization",
            "required": true,
            "type": "FieldAutoComplete",
            "multiple": false,
            "chooseClassif": {
                "classif": "ORGANIZ_CL",
                "multi": false,
                "returnDue": true
            }
        },
        {
            "name": "RegistrationOnly",
            "type": "FieldCheckbox",
            "description": "Отправлять только при регистрации РК"
        },
        {
            "label": "Тип",
            "name": "Type",
            "type": "FieldEnum",
            "required": true,
            "options": [
                {
                    "value": 1,
                    "label": "Оригинал"
                },
                {
                    "value": 2,
                    "label": "Любой экземпляр"
                },
                {
                    "value": 3,
                    "label": "Копия"
                }
            ]
        }
    ],
    {
        "label": "Метод авторизации",
        "name": "AuthorizationMethod",
        "type": "FieldEnum",
        "required": true,
        "options": [
            {
                "value": "NONE",
                "label": "Авторизация не требуется"
            },
            {
                "value": "LOGIN",
                "label": "Авторизация логин/пароль"
            },
            {
                "value": "NTLM",

```

```

        "label": "Авторизация Windows"
    }
],
"onChange": [
    {
        "action": "disableField",
        "field": "SmtpUsername",
        "value": "NONE"
    },
    {
        "action": "disableField",
        "field": "SmtpPassword",
        "value": "NONE"
    },
    {
        "action": "enableField",
        "field": "SmtpUsername",
        "value": "LOGIN"
    },
    {
        "action": "enableField",
        "field": "SmtpPassword",
        "value": "LOGIN"
    },
    {
        "action": "enableField",
        "field": "SmtpUsername",
        "value": "NTLM"
    },
    {
        "action": "enableField",
        "field": "SmtpPassword",
        "value": "NTLM"
    }
]
},
{
    "label": "Типы",
    "name": "Types",
    "type": "FieldEnum",
    "enumType": "multiple",
    "options": [
        {
            "value": 1,
            "label": "Тип 1"
        },
        {
            "value": 2,
            "label": "Тип 2"
        },
        {
            "value": 3,
            "label": "Тип 3"
        }
    ]
},
{
    "label": "Начиная с даты",
    "name": "StartDate",
    "type": "FieldDateTime",
    "minDate": "2021",
    "maxDate": "05.25.22"
},

```

```

{
  "label": "Название папки",
  "name": "Folder",
  "type": "FieldText",
  "required": true
},
{
  "label": "Время",
  "name": "Time",
  "required": true,
  "type": "FieldTimePicker"
},
{
  "label": "Пароль",
  "name": "Password",
  "type": "FieldPassword",
  "required": true
},
{
  "label": "Профиль эл. почты",
  "name": "EmailProfile",
  "type": "FieldProfile",
  "required": true,
  "selectProfile": {
    "namespace": "Eos.Platform.Settings.Email",
    "typename": "CommonCfg",
    "propertyProfileName": "ProfileName"
  }
},
{
  "label": "Профиль криптографии",
  "name": "CryptoProfile",
  "type": "FieldProfile",
  "required": true,
  "selectProfile": {
    "namespace": "Eos.Delo.Settings.Cryptography",
    "typename": "CryptographyCfg",
    "propertyProfileName": "ProfileName"
  }
},
{
  "label": "Хранилище",
  "name": "Storage",
  "type": "FieldStorage",
  "required": true
}
],
"rows": [
  {
    "cells": [
      {
        "fields": [ "Organization" ],
        "width": 24
      }
    ]
  },
  {
    "cells": [
      {
        "fields": [ "RegistrationOnly" ],
        "width": 24
      }
    ]
  }
]

```

```

},
{
  "cells": [
    {
      "fields": [ "Type" ],
      "width": 24
    }
  ]
},
{
  "cells": [
    {
      "fields": [ "AuthorizationMethod" ],
      "width": 24
    }
  ]
},
{
  "cells": [
    {
      "fields": [ "Types" ],
      "width": 24
    }
  ]
},
{
  "cells": [
    {
      "fields": [ "StartDate" ],
      "width": 24
    }
  ]
},
{
  "cells": [
    {
      "fields": [ "Storage" ],
      "width": 12
    },
    {
      "fields": [ "Folder" ],
      "width": 12
    }
  ]
},
{
  "cells": [
    {
      "fields": [ "Time" ],
      "width": 24
    }
  ]
},
{
  "cells": [
    {
      "fields": [ "Password" ],
      "width": 24
    }
  ]
},
{
  "cells": [

```

```

        {
            "fields": [ "EmailProfile" ],
            "width": 24
        }
    ]
},
{
    "cells": [
        {
            "fields": [ "CryptoProfile" ],
            "width": 24
        }
    ]
}
],
"defaultValues": {
    "Folder": "NewFolder",
    "Type": 1,
    "AuthorizationMethod": "NONE",
    "Types": [ 1, 2 ],
    "StartDate": "2022.16.04",
    "Time": "13:55"
}
}
}

```

Объект "rows" содержит список строк, в данном примере в общих настройках одна строка, в экземплярных – несколько.

Объект "cells" – содержит список столбцов. Каждый столбец содержит список полей "fields", и ширину "width". При "width": 24 – столбец занимает всю строку, "width":12 – половину строки, "width":6 – четверть строки.

Например, чтобы поле выбора хранилищ "Storage", и название папки "Folder" находились в одной строке, двух соседних столбцах и занимали по половине строки:

```

"rows": [
    { "cells": [{"fields": ["Storage"], "width": 12}, {"fields": ["Folder"], "width": 12}] }
]

```

Регистрируемые классы конфигураций для примера выше:

```

using Eos.Platform.AppSettings.Abstractions;
using System.Collections.Generic;
namespace Eos.Delo.Settings.TestDaemons
{
    public class CheckParamsDefaultCfg
    {
        public long Periodicity { get; init; }
    }

    public class CheckParamsJobCfg
    {
        public string Organization { get; init; }
        public bool RegistrationOnly { get; init; }
        public long Type { get; init; }
        public string AuthorizationMethod { get; init; }
        public IReadOnlyList<long> Types { get; init; }
        public string StartDate { get; init; }
        public string Folder { get; init; }
        public string Time { get; init; }
        public Secret Password { get; init; }
        public string EmailProfile { get; init; }
        public string CryptoProfile { get; init; }
        public string Storage { get; init; }
    }
}

```

```
}
```

Пример получения параметров с получением значения пароля:

```
protected override async Task FillParamsAsync()
{
    commonCfg = GetDefaultAppSettingsParams<CheckParamsDefaultCfg>();
    jobCfg = GetJobAppSettingsParams<CheckParamsJobCfg>();
    await InContextAsync(async (context, serviceProvider) => {
        var secretStore = serviceProvider.GetRequiredService<ISecretStore>();
        _password = await secretStore.GetValueAsync(jobCfg.Password);
    });
}
```

Примеры конфигурационных файлов также можно посмотреть в дистрибутиве системы в каталоге «Sdk» (проект Eos.Delo.Samples.Daemons, папка Setup).

6.4.2 Добавление metadata.json

В проекте, содержащем фоновую задачу, необходимо создать файл "metadata.json" в кодировке UTF-8, с описанием модуля. В свойствах файла необходимо указать значение свойства «Копировать в выходной каталог» как «Всегда копировать» («Copy to Output Directory» - «Copy always»).

Пример файла metadata.json:

```
{
  "Name": "Eos.Delo.Daemons.TestDaemon.Module",
  "Version": "1.0.0"
}
```

В "Name" указывается название модуля, соответствует названию собираемой библиотеки (проекта).

Также, необходимо подготовить файл с названием подключаемого модуля с расширением ".json", в данном случае "Eos.Delo.Daemons.TestDaemon.Module.json". Содержание файла соответствует файлу "metadata.json".

Данные файлы нужны для регистрации модуля на хосте.

6.4.3 Загрузка фоновой задачи в систему

Изменение списка загружаемых модулей, удаление или добавление модуля, производится на остановленном пуле хоста (delosrv). После необходимых изменений хост запускается.

Загрузка сборок (библиотек) фоновых задач может осуществляться в папку модуля в lib\appsrv. Библиотека фоновой задачи, файлы сборки, файл "metadata.json" помещаются в папку ...\lib\appsrv\ModuleName, где ModuleName соответствует имени модуля из "metadata.json". Для примера из п. 6.4.2 такой папкой будет ...\lib\appsrv\Eos.Delo.Daemons.TestDaemon.Module.

Данная папка имеет содержимое (см. Рисунок 5):

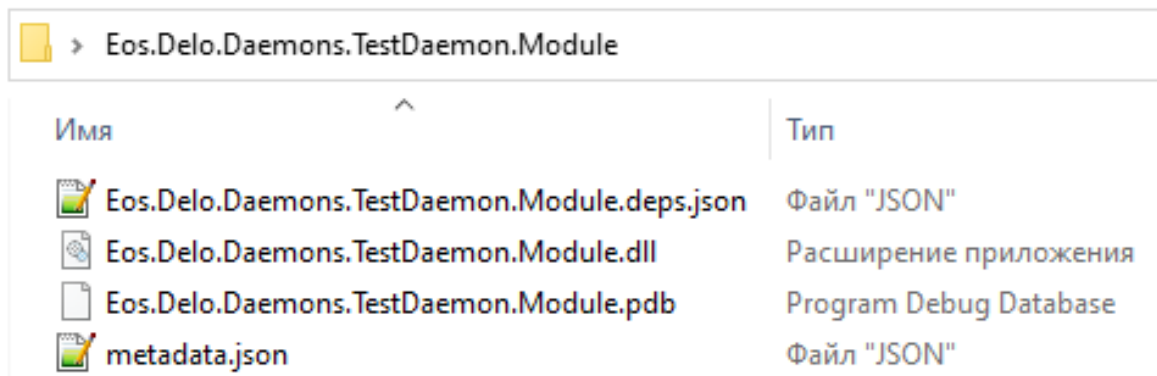


Рисунок 5

Файлы:

Eos.Delo.Daemons.TestDaemon.Module.deps.json - содержит информацию о зависимостях проекта

Eos.Delo.Daemons.TestDaemon.Module.dll - библиотека фоновой задачи

Eos.Delo.Daemons.TestDaemon.Module.pdb - файл с отладочными данными (формируется только при сборке в конфигурации Debug, если указано его формирование) - может отсутствовать

metadata.json - файл с описанием модуля

Для инициализации модуля необходимо добавить ранее подготовленный файл с названием модуля + ".json" из п. 6.4.2 в папку ...\\config*-delosrv\\Settings. Если папка модуля находится не в lib\\appsrv, в данном файле "*.json" необходимо указать путь в ключе "Path" файла.

Пример "Eos.Delo.Daemons.TestDaemon.Module.json" с указанием "Path":

```
{
  "Name": "Eos.Delo.Daemons.TestDaemon.Module",
  "Version": "1.0.0",
  "Path": "C:\\CustomModules\\Eos.Delo.Daemons.TestDaemon.Module"
}
```

После загрузки библиотек необходимо запустить хост.

6.4.4 Настройка фоновой задачи

Управление фоновыми задачи происходит в UI управления фоновых задач. Чтобы в него перейти, с главной страницы Дело-WEB, пункт меню "Администрирование" → Справочники → Сервисы → Управление фоновыми задачами (см. Рисунок 6).

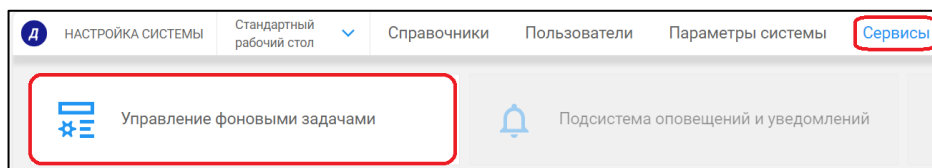


Рисунок 6

После добавления библиотек на хост, во вкладке "Конфигуратор" необходимо добавить домен(ы), подключить к нему(ним) хост(ы) и назначить фоновые задачи. Назначение фоновых задач производится на выбранный домен (не хост) по кнопке "Назначить на домен" из правой панели (см. Рисунок 7).

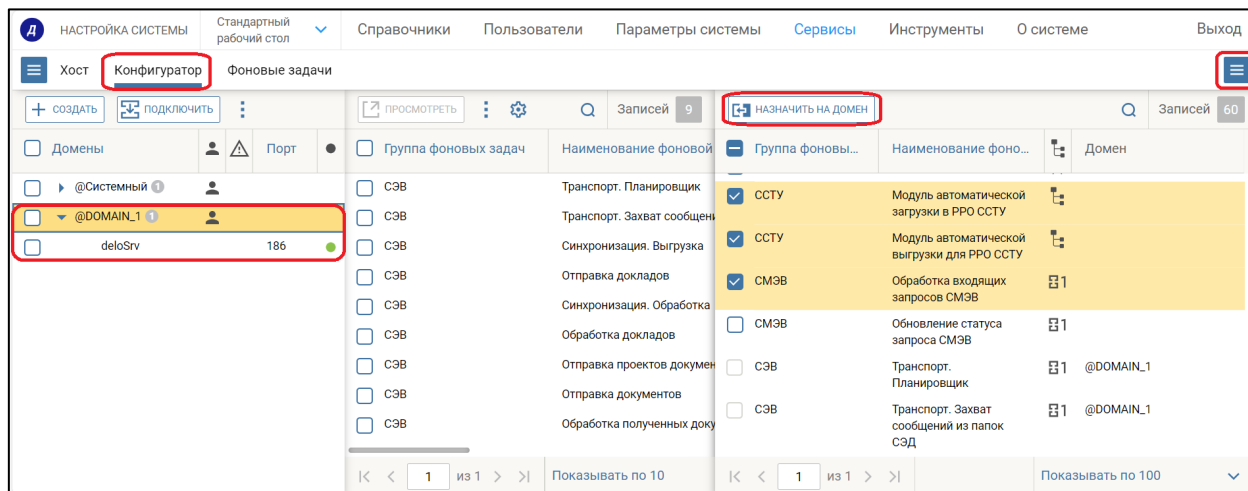


Рисунок 7

Внимание!

Список для назначения домены формируется из последних конфигураций всех хостов. Поэтому, рекомендуется поддерживать одинаковый состав ФЗ на хостах. В ином случае, если к домену подключен хост1, а назначены ФЗ, имеющиеся только на хост2, экземпляры таких ФЗ для домена не будут созданы и во вкладке "Фоновые задачи" будут отсутствовать записи по ним.

Для исключения фоновых задач с домена, их нужно выбрать и нажать на кнопку "Исключить". Выбранные ФЗ будут исключены с домена и удалены их конфигурации (см. Рисунок 8).

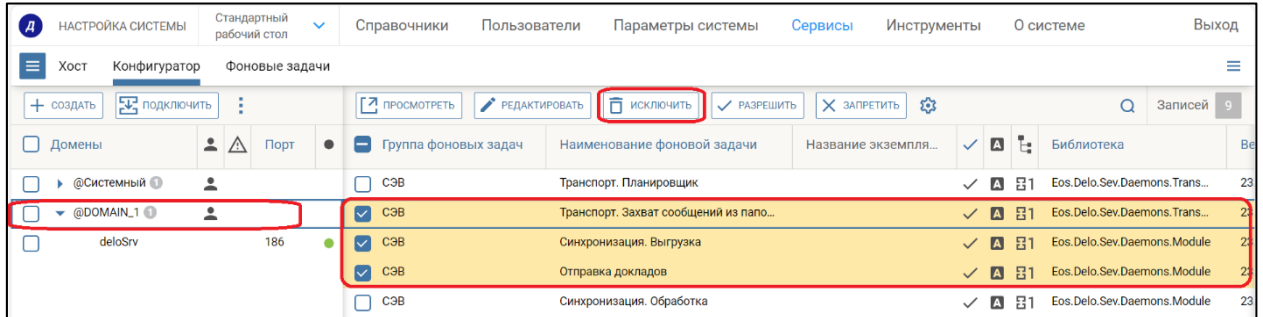


Рисунок 8

Для переключения ФЗ (выбранного экземпляра) на другой домен, необходимо нажать на кнопку редактирования ФЗ (см. Рисунок 9), выбрать домен и нажать на кнопку сохранения (см. Рисунок 10).

Чтобы ФЗ начала работу, на домене должны быть подключены запущенные хосты, и ФЗ разрешена к запуску. Количество подключенных хостов (без учета статуса хоста) указано в правой части наименования домена.

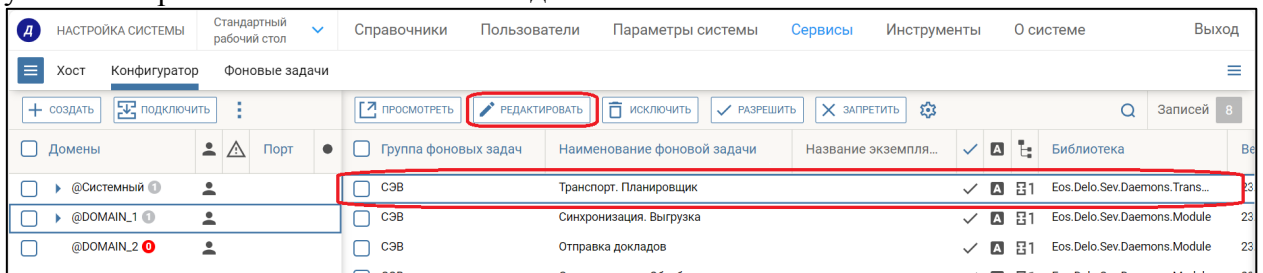


Рисунок 9

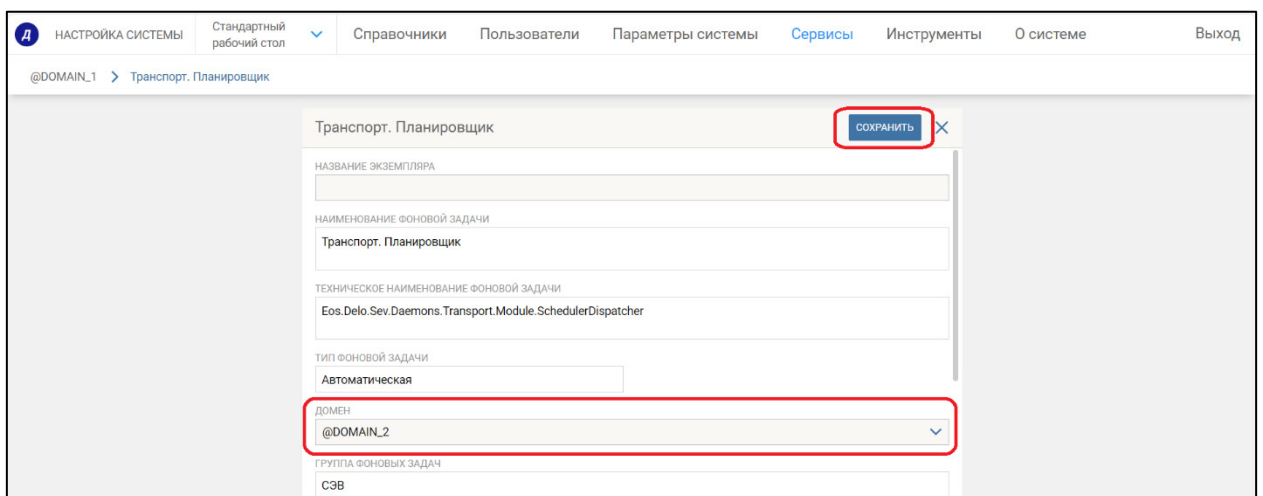


Рисунок 10

После назначения, все разрешенные к запуску фоновые задачи, отобразятся в списке фоновых задач во вкладке «Фоновые задачи».

ФЗ не запустится, если она требует настройки и не настроена на данный момент.

Для настройки ФЗ нужно выделить ФЗ для настройки и нажать на кнопку "Настроить задачу". После чего появится окно настройки ФЗ, в котором нужно будет заполнить соответствующие параметры настроек задачи и нажать "Сохранить". После сохранения, фоновая задача будет запущена (см. Рисунок 11). Конфигурация сохраняется в таблице APP_SETTINGS.

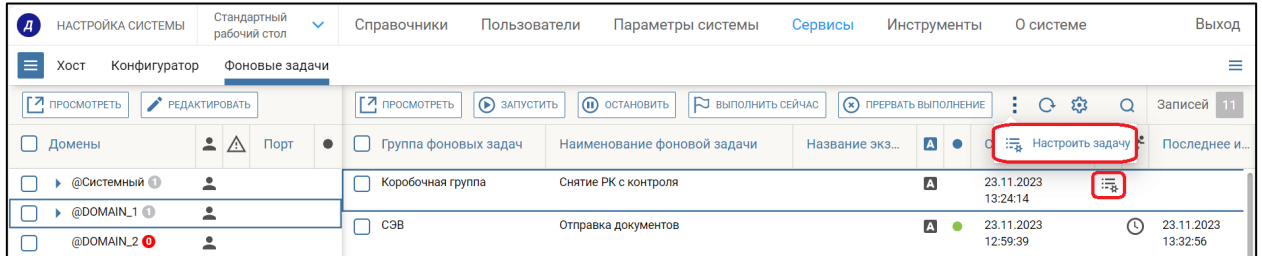


Рисунок 11

6.5 Создание спец. событий для потребления ФЗ

В случае, когда по требованию пользователя необходимо вызвать обработку определенной РК или РКПД фоновой задачей, работающей в автоматическом режиме, можно использовать добавление специальных событий для потребления ФЗ.

Фоновая задача должна быть реализована с отбором событий с указанными флагами.

Необходимо создать и добавить кнопку-плагин на РК/РКПД, при нажатии на которую будет происходить вызов SOp (ДЕЛО API 2009) или метод контроллера GraphQL (ДЕЛО API 2023) и добавление специального события. Документация на GraphQL описана в разделе [4 GraphQL \(для ДЕЛО API 2023\)](#).

После чего, ФЗ, отобрав события, обработает данные РК/РКПД.

Для добавления специального события используется:

SOp AddEvent с аргументами / мутация addEvtQueueItem GraphQL с параметрами:

- kind_event / kindEvent - тип события. Может принимать значения: 1 - создание, 2 - изменение.
- object_name / objectName - тип объекта, имя равно значению поля object_name таблицы eos_object. Например, DOC_RC для РК, PRJ_RC для РКПД.
- object_id / objectId - id объекта, isn РК или РКПД.
- flags - строка, содержащая спец. флаг, по которому ФЗ будет отбирать события. Если спец. флагов несколько, они должны быть записаны через запятую, без пробелов. Например, "SPEC_FLAG1,SPECIAL_FLAG2".
- due_docgroup / dueDocgroup - due группы документов, заполняется для РК/РКПД/Резолюций.

Значение аргумента flags не должно содержать стандартных флагов, используемых в системе «ДЕЛО», т.к. создание таких событий может привести к неправильной работе системы.

Пример вызова SOp:

В js скрипте плагина:

```
// Пример вызова СОП с подставленными данными
ds.loadT({ AddEvent: args({ kind_event: 2, flags: "SPEC_FLAG1,SPECIAL_FLAG2",
due_docgroup: "0.2TZ.2U1.", object_id: "4333", object_name: "DOC_RC" })
}).promise()
.then(function (data) {deloAlert(data.result)},
function (err) {deloAlert("Ошибка добавления события")});
//Равносильно GET запросу
.../CoreHost/OData/AddEvent?kind_event=2&flags="SPEC_FLAG1,SPECIAL_FLAG2"&du
```

```
e_docgroup="0.2TZ.2U1."&object_id="4333"&object_name="DOC_RC"&$format=compact

// Пример запроса с PK
ds.loadT({ AddEvent: args({ kind_event: 2, flags: "SPEC_FLAG", due_docgroup:
pm.DOC_RC.DUE_DOCGROUP, object_id: pm.DOC_RC.ISN_DOC, object_name: "DOC_RC" })
}).promise()
.then(function (data) {deloAlert(data.result)},
function (err) {deloAlert("Ошибка добавления события")});
//Равносильно GET запросу со значением параметров из PK
../../CoreHost/OData/AddEvent?kind_event=2&flags="SPEC_FLAG"&due_docgroup="0.2
TZ.2U1."&object_id=4333&object_name="DOC_RC"&$format=compact
```

Пример вызова контроллера GraphQL:

```
В js скрипте:
// Пример с подставленными данными
const docRcParms = {
  Isn: 4333,
  Docgroup: "0.2TZ.2U1."
}
//Для запроса с PK параметры аналогичные docRcParms: docRcParms.Isn =
pm.DOC_RC.ISN_DOC, docRcParms.Docgroup = pm.DOC_RC.DUE_DOCGROUP
let mutation = {
  query:
    `mutation {
      addEvtQueueItem(
        input: {
          clientMutationId: "addEvtQueueItem_1",
          kindEvent: 2,
          objectName: "DOC_RC",
          objectId: "${docRcParms.Isn}",
          dueDocgroup: "${docRcParms.Docgroup}",
          flags: ",SPEC_FLAG,"
        })
      {
        clientMutationId
        success
        message
        messageCode
        messageData
        systemMessage
      }
    }`
}
$.ajax({
  type: "POST",
  contentType: "application/json;",
  accept: "application/json;",
  dataType: "json",
  data: JSON.stringify(mutation),
  url: rootpath + "CoreHost/gql/query",
  success: res => {
    if (res.data) {
      if (res.data.addEvtQueueItem.success)
        deloAlert("Событие успешно добавлено")
      else deloAlert(`${res.data.addEvtQueueItem.message}`)
    }
    else deloAlert("Ошибка добавления события")
  },
  error: err => {
    deloAlert("Ошибка добавления события")
  }
})
```

Если требуется работать с ФЗ по запросам пользователей с передачей каких-либо данных, когда требует длительное время обработки запроса, можно воспользоваться записью параметров в таблицу CUSTOM_STORAGE, из которой ФЗ считывает информацию.

Фоновая задача должна быть реализована на поиск и чтение записи из таблицы CUSTOM_STORAGE и после выполнения обработки запись удалять, либо модифицировать, в зависимости от требований к работе данной ФЗ. Если записи в таблице CUSTOM_STORAGE после отработки ФЗ не будут использоваться, их необходимо удалять, чтобы не перезагружать таблицу лишними данными.

Записи в CUSTOM_STORAGE должны содержать уникальный идентификатор valueId (строка), ownerId (строка) - идентификатор объекта владельца записи, value (строка) - значение, для считывания фоновой задачи.

Запросы на данную таблицу с использованием оператора LIKE могут нагружать БД, поэтому рекомендуется использовать при поиске не только valueId, но также ownerId и ownerKind.

Столбец value ограничен значением в 2000 символов. Для записи более длинных строк, необходимо разделять value на строки длиной не более 2000 символов и добавлять записями в таблицу с ordernum от 0 до n с одинаковым valueId.

Пример вызова контроллера GraphQL:

В js скрипте:

```
let mutation = {
  query:
    `mutation {
      createCustomStorage(
        input: {
          clientMutationId: "createCustomStorage_1",
          data:{
            valueId:"CustomNameForMosule.Eos.Delo.Samples.FZ#1",
            ordernum: 0,
            value: "command#1#${pageContext.CurrentUser.ISN_LCLASS
IF}",
            ownerId:"Eos.Delo.Samples.FZ"
          }
        }
      )
    }
    {
      clientMutationId
      success
      message
      messageCode
      messageData
      systemMessage
    }
  }`
}
$.ajax({
  type: "POST",
  contentType: "application/json;",
  accept: "application/json;",
  dataType: "json",
  data: JSON.stringify(mutation),
  url: rootpath + "CoreHost/gql/query",
  success: res => {
    if (res.data) {
      if (res.data.createCustomStorage.success)
        deloAlert("Запись успешно добавлена")
      else
        deloAlert(`${res.data.createCustomStorage.message}`)
    }
    else deloAlert("Ошибка добавления записи")
  }
})
```

```

    },
    error: err => {
        deloAlert("Ошибка добавления записи")
    }
})

```

Примером работы ФЗ с поиском записей в CUSTOM_STORAGE является выгрузка расширенного протокола. Пользователь нажимает на кнопку выгрузки, посылается запрос с указанием количества выгружаемых записей протокола за итерацию обработки одного запроса в ФЗ. Фоновая задача с определенным интервалом времени опрашивает таблицу на наличие нужных ей записей. Если находит запрос на выгрузку, выполняет его, записывая промежуточные точки выполнения в эту же таблицу, чтобы при сбое работы сервера была возможность продолжить начатую работу, а не начинать сначала. После выполнения выгрузки запись в CUSTOM_STORAGE с промежуточными точками выполнения и запросом удаляется, и записывается новая, что запрос успешно выполнен. Последняя запись считывается на определенной странице web и информирует пользователя об успешном завершении выгрузки. Такая выгрузка может длиться несколько часов, поэтому организована в работе через фоновую задачу, а не с помощью кастомного контроллера, которые используются для быстрых операций.

Запросы с web напрямую не уходят в фоновую задачу. Для подобного взаимодействия используются события и записи в таблицу CUSTOM_STORAGE.

Внимание!

Фоновые задачи должны работать только со своими записями в CUSTOM_STORAGE (создавать/модифицировать/удалять) с уникальными value_id. В противном случае, если удалить или модифицировать чужую запись, будет нарушена работа других ФЗ.

6.6 Тестирование

Для тестирования методов разрабатываемых проектов фоновых задач на существующей базе данных необходимо выполнить следующие шаги:

Создать Nunit Test project (см. Рисунок 12)

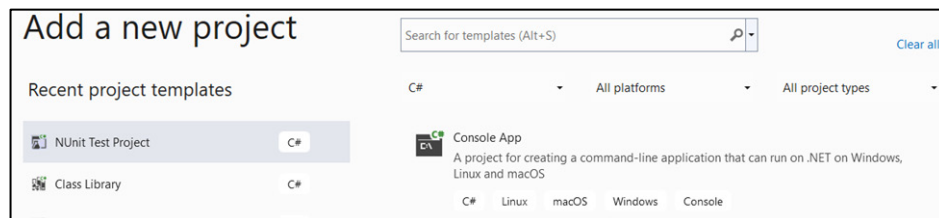


Рисунок 12

Подключить NuGet пакеты

- Eos.Delo.Platform.Storage.Model
- Eos.Delo.Platform.Storage.Tests.Base

Пример подключения зависимостей <ItemGroup>:

```

<PackageReference Include="Eos.Delo.Platform.Storage.Model" Version="23.5.*" />
</PackageReference>
<PackageReference Include="Eos.Delo.Platform.Storage.Tests.Base" Version="23.5.*" />
</PackageReference>
<PackageReference Include="Microsoft.AspNetCore.Mvc.Testing" Version="5.0.11" />
</PackageReference>
<PackageReference Include="nunit" Version="3.13.2" />
<PackageReference Include="NUnit3TestAdapter" Version="4.0.0" />
<PackageReference Include="Microsoft.NET.Test.Sdk" Version="17.0.0" />

```

Добавить файл appsettings.json

Для подключения к базе данных используется информация из файла appsettings проекта. В нем настраивается уровень логгирования, строки подключения, информация о приложении. Файл appsettings.json добавляется в проект с дополнительным свойством **"Копировать в выходной каталог" - "Всегда"**:

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Debug",
      "System": "Information",
      "Microsoft": "Information"
    }
  },
  "Eos": {
    "DynamicAssemblyPath": "dynamic-assemblies",
    "DynamicMetadataPath": "ar-metadata"
  },
  "ConnectionStrings": {
    "FzName.Test:SQLServer": "Server=.\sqlexpress;User
ID=delo;Password=delo;Initial Catalog=DELO_DB",
    "FzName.Test:PostgreSQL": "Host=localhost;Port=5432;Database=delo;
User ID=delo;Password=delo;CommandTimeout=20;Timeout=40;",
    "FzName.Test:Oracle": "Data Source=localhost:1521;User
Id=delo;Password=delo"
  },
  "ApplicationInfo": {
    "Name": "TestApp",
    "AppVersion": "1.0"
  }
}
```

Описание строки подключения к БД можно взять из настроек тестового стенда в файле appsettings.json по ключу "ConnectionStrings.Default".

Создать класс TestSetup

Для подключения к БД необходимо создать класс, наследуемый от TestsSetupGql:

```
// Создание класса, наследуемого от TestsSetupGql
using Eos.Delo.Platform.Storage.Tests.Base;
using Eos.Platform.Metadata;

namespace FzName.Test
{
    public class TestsSetup : TestsSetupGql
    {
        public TestsSetup(HbmProviderType providerType =
HbmProviderType.SQLServer) : base(providerType) { }
    }
}
```

Указать namespace в котором содержится класс TestSetup для строк connectionString в файле appsettings.json перед типом СУБД ("FzName.Test:SQLServer").

Создать класс с тестами.

В классе тестов необходимо создать метод OneTimeSetUp, который будет выполняться перед запуском тестов, конфигурирующий провайдер подключения к БД.

```
// Создание класса с тестами
namespace FzName.Test
{
    public class Tests
    {
```

```

private Dictionary<HbmProviderType, TestsSetup> _testsSetup = new()
{
    { HbmProviderType.SQLServer, new
TestsSetup(HbmProviderType.SQLServer) },
    { HbmProviderType.PostgreSQL, new
TestsSetup(HbmProviderType.PostgreSQL) },
    { HbmProviderType.Oracle, new TestsSetup(HbmProviderType.Oracle) }
};

[OneTimeSetUp]
public virtual void OneTimeSetUp()
{
    foreach (var setup in _testsSetup.Values)
    {
        //здесь можно добавить свои сервисы в ServiceCollection и
        //если указан параметр isMainData = true то будут выполнены
        файлы конфигурации в configurationBuilder
        //если указан параметр isMainData = true то будут выполнены
        скрипы Data/{HbmProviderType}/adddata.sql и
        Data/{HbmProviderType}/{dataSql[i]}.sql
        setup.OneTimeSetUpBase(new ServiceCollection(), services =>
        {
            }, builder => { }, isCustomQueries: true,
            isMainData: false, aliasName: "", useArMetadata: true,
            useLogging: true, arMetadataGeneration: true, dataSql: new
string[0]);

        if (setup.ProviderNotConfigured)
            throw new Exception();
    }
}

[Test]
[TestCase(HbmProviderType.SQLServer)]
[TestCase(HbmProviderType.Oracle)]
public void Test1(HbmProviderType providerType)
{
    var db =
_testsSetup[providerType].ServiceProvider.GetRequiredService<IDataContext>();

    //логика теста
}
}
}

```

Пример тестов для ФЗ находится в дистрибутиве системы в каталоге «Sdk» (проект Eos.Delo.Samples.Daemons.UnitTest).

При необходимости работы с доп. реквизитами, перед их получением необходимо добавить вызов метода ArValuesGenerateSuccess

Код ArValuesGenerateSuccess

```

// Дополнительные реквизиты - добавление вызова метода ArValuesGenerateSuccess
private void ArValuesGenerateSuccess(HbmProviderType providerType)
{
    using var scope = MutationSetup.ServiceProvider.CreateScope();
    var localProvider = scope.ServiceProvider;
    var configuration = localProvider.GetRequiredService<IConfiguration>();
    var connectionString = configuration.GetConnectionString("FzName.Test:"
+ providerType.ToString());
    var path = configuration.GetDynamicMetadataPath();
    if (string.IsNullOrEmpty(connectionString))
        throw new InvalidOperationException($"строка соединения

```



```

FzName.Test:{providerType} не найдена в конфигурационном файле.");
    ArValueXmlMetadataGenerationHelper.Generate(path, connectionString,
providerType);

    Assert.IsTrue(File.Exists(Path.Combine(path, "ArValuesAdd.hbm.xml")));

    var stream =
typeof(HbmClass).Assembly.GetManifestResourceStream("Eos.Platform.Metadata.eos
-platform-mapping.xsd");

    string readXsdMessage = "";
    var schema = XmlSchema.Read(stream, (o, a) => {
        Assert.Warn(a.Message);
        readXsdMessage += a.Message;
    });

    Assert.IsTrue(string.IsNullOrEmpty(readXsdMessage));

    var schemas = new XmlSchemaSet();
    schemas.Add(schema);

    string validateMessage = "";
    var xDoc = XDocument.Load(Path.Combine(path, "ArValuesAdd.hbm.xml"));
    xDoc.Validate(schemas, (o, a) => {
        Assert.Warn(a.Message);
        validateMessage += a.Message;
    });

    Assert.IsTrue(string.IsNullOrEmpty(validateMessage));
}

```

Возможные ошибки:

- Если тесты не запускаются, необходимо проверить и обновить зависимости.
- Если появляется ошибка "error CodeGeneration: Error: Связанный тип ... не найден», следует обновить версии Eos.Delo.Platform.Storage.Model и Eos.Delo.Platform.Storage.Tests.Base.
- Если ошибка "System.InvalidCastException: Unable to cast object of type 'Eos.Delo.Platform.Storage.Model.DataContextCaching' to type 'Eos.Platform.AppSettings.IAppSettingsContext'".

Необходимо добавить в файл appsettings.json следующий код с названием модуля, проекта тестов:

```

"Eos": {
  "DynamicAssemblyPath": "dynamic-assemblies",
  "DynamicMetadataPath": "ar-metadata",
  "ExtensionDbContexts": {
    "Eos.Platform.AppSettings.IAppSettingsContext": {
      "Module": "FzName.Test",
      "Library": "Eos.Platform.AppSettings"
    },
    "Eos.Platform.Licensing.ILicenseContext": {
      "Module": "FzName.Test",
      "Library": "Eos.Platform.Licensing"
    }
  }
}

```

7 Форматы обмена данными

7.1 Паспорт РК электронного сообщения E-mail

Настоящий раздел описывает формат файла паспорта РК электронного сообщения E-mail, обеспечивающего информационное взаимодействие между различными экземплярами СЭД «ДЕЛО», а также взаимодействие СЭД «ДЕЛО» с другими приложениями.

Паспорт РК оформляется как самостоятельный XML-документ с корневым элементом passport, файл паспорта РК имеет фиксированное имя - passport.xml.

Уведомление о регистрации оформляется как самостоятельный XML-документ с корневым элементом receipt, файл паспорта РК имеет фиксированное имя - receipt.xml.

Уведомление об отказе от регистрации оформляется как самостоятельный XML-документ с корневым элементом refusal, файл паспорта РК имеет фиксированное имя - refusal.xml.

Содержимое паспорта РК делится на несколько зон.

Ниже приведено описание зон и элементов паспорта и уведомлений, соответствующие им XSD-схемы приведены в Приложение К.

7.1.1 Перечень и содержание зон сообщения

Перечень и содержание зон сообщения представлены ниже (см. Таблица 6).

Таблица 6

Наименование зоны сообщения	Имя типа элемента XML-документа	Содержание зоны сообщения
Заголовок	header	Содержит служебную информацию, необходимую для правильной передачи и интерпретации всего сообщения в целом. Обязательная зона для всех видов сообщений.
Документ (основная зона)	document	Содержит информацию о реквизитах пересылаемого документа, а также информацию о выданных заданиях на исполнение и обработку документа (в виде резолюций и поручений). Обязательная зона для сообщений о пересылке документа
Расширение	Expansion	Содержит дополнительные, не вошедшие в основную зону, данные из системы отправителя.
Уведомление (подтверждение приема)	Acknowledgement	Содержит ответную информацию о доставке сообщения, об ошибках приема и интерпретации сообщения, о регистрации полученного документа и др. Обязательная зона для сообщений вида уведомление.

7.1.2 Правила описания зон сообщения

Зоны сообщения оформляются как самостоятельные, независимые элементы XML-файла первого уровня иерархии. Передаваемая в зоне сообщения информация оформляется как вложенные элементы XML-файла второго и последующих уровней иерархии.

Как и для элемента – описателя зоны сообщения, так и у вложенных элементов указывается их имя типа, содержание и допустимые атрибуты.

При описании вложенных элементов дополнительно указывается их допустимое количество (кратность):

- 1 - обязательный, может встречаться только один раз;
- 1-n - обязательный и может встречаться несколько раз;
- 0-1 – не обязательный, может встретиться только один раз;

- 0-n – не обязательный, может встречаться несколько раз.
- Для атрибутов элементов кратность может быть установлена только равной:
- 1 – обязательный атрибут;
- 0 – не обязательный атрибут.
- При описании атрибутов и содержания элементов используются следующие типы данных:

- String — строка;
- Date — дата в формате уууу-mm-dd, где уууу – обозначение года, mm – обозначение месяца, dd – обозначение дня;
- DateTime — дата и время в формате уууу-mm-ddThh:mm:ss, где уууу – обозначение года, mm – обозначение месяца, dd – обозначение дня, T – разделитель даты и времени, hh - обозначение часа, mm – обозначение минуты, ss – обозначение секунды;
- Num — целое число;
- Enum — целое число из перечня допустимых значений.

7.1.3 Описание зоны "Заголовок"

Зону сообщения "Заголовок" образует первого уровня иерархии — header, определяющий версию стандарта, по которому сформировано сообщение, вид передаваемого сообщения, дату и время его формирования и от кого(кому) и когда оно передается.

Зона состоит из одного элемента header.

7.1.4 Описание зоны "Документ"

Зону сообщения "Документ" образует элемент document первого уровня иерархии и вложенные в него элементы.

Зона сообщения "Документ" содержит значения реквизитов пересылаемого документа, а также поручений.

Наименования и уровни вложенности элементов, входящих в состав зоны сообщения "Документ", при передаче полной информации о документе приведены ниже. (см. Таблица 7):

Таблица 7

Наименования и уровни вложенности элементов зоны "Документ"						Кратность	Комментарий
1	2	3	4	5	6		
document	-	-	-	-	-	0-1	Кратность = 0 для сообщений вида уведомление
-	RegNumber	-	-	-	-	1	Содержит информацию о регистрации документа в отправляющей системе
-	Confident	-	-	-	-	1	Содержит информацию о грифе документа
-	Author	-	-	-	-	0-n	Содержит информацию об авторе документа ³

³ При пересылке входящих документов, включая письма граждан, содержит информацию о корреспонденте этих документов.

Наименования и уровни вложенности элементов зоны "Документ"						Кратность	Комментарий
1	2	3	4	5	6		
-	-	Organization	-	-	-	0-1	Кратность = 1 для авторов – организаций (юридических лиц)
-	-	-	Address	-	-	0-1	
-	-	-	Econtact	-	-	0-n	
-	-	-	OfficialPerson	-	-	0-n	Содержит информацию о должностном лице, подписавшем документ
-	-	-	-	Name	-	1	
-	-	-	-	Official	-	0-1	
-	-	-	-	SignDate	-	0-1	
-	-	PrivatePerson	-	-	-	0-1	Кратность = 1 для авторов - физических лиц
-	-	-	Name	-	-	1	
-	-	-	Address	-	-	0-1	
-	-	-	Econtact	-	-	0-n	
-	-	OutNumber	-	-	-	0-1	Содержит информацию о регистрации документа в организации автора документа ⁴
-	-	-	RegNumber	-	-	1	
-	Validator	-	-	-	-	0-n	Содержит информацию о визировании документа в организации - отправителе
-	-	Organization	-	-	-	1	
-	-	-	Address	-	-	0-1	
-	-	-	Econtact	-	-	0-n	
-	-	-	OfficialPerson	-	-	0-n	
-	-	-	-	Name	-	1	
-	-	-	-	Official	-	0-1	
-	-	-	-	SignDate	-	0-1	
-	Addressee	-	-	-	-	0-n	Содержит информацию об адресатах документа

⁴ При отправке исходящего документа дублирует информацию, содержащуюся в элементе Document / RegNumber.

Наименования и уровни вложенности элементов зоны "Документ"						Кратность	Комментарий
1	2	3	4	5	6		
-	-	Organization	-	-	-	0-1	Кратность = 1 для адресатов – организаций (юридических лиц)
-	-	-	Address	-	-	0-1	
-	-	-	Econtact	-	-	0-n	
-	-	-	OfficialPerson	-	-	0-n	Содержит информацию о должностном лице – адресате документа
-	-	-	-	Name	-	1	
-	-	-	-	Official	-	0-1	
-	-	PrivatePerson	-	-	-	0-1	Кратность = 1 для адресатов - физических лиц
-	-	-	Name	-	-	1	
-	-	-	Address	-	-	0-1	
-	-	-	Econtact	-	-	0-n	
-	Writer	-	-	-	-	0-1	Содержит информацию об исполнителе документа в организации - отправителе
-	-	Organization	-	-	-	1	
-	-	-	Address	-	-	0-1	
-	-	-	Econtact	-	-	0-n	
-	-	-	OfficialPerson	-	-	0-n	
-	-	-	-	Name	-	1	
-	-	-	-	Official	-	0-1	
	Task	-	-	-	-	0-n	
	-	TaskNumber	-	-	-	1	Номер пункта документа или дата резолюции
	-	Confident	-	-	-	1	Признак конфиденциальности и поручения
	-	Referred	-	-	-	1-n	Ссылка на родительский объект: Документ или поручение
	-	-	RegNumber	-	-	0-1	Обязательно

Наименования и уровни вложенности элементов зоны "Документ"						Кратность	Комментарий
1	2	3	4	5	6		
	-	-	TaskNumber	-	-	0-1	заполняется либо элемент RegNumber (ссылка на документ), либо элемент TaskNumber (ссылка на поручение)
	-	Author	-	-	-	0-1	Заполняется только для резолюций и содержит информацию о ее авторе
	-	-	Organization	-	-	1	Содержит информацию об организации автора резолюции
	-	-	-	Address	-	0-1	
	-	-	-	Econtact	-	0-n	
	-	-	-	OfficialPerson	-	0-n	Содержит информацию о должностном лице – авторе резолюции
	-	-	-	-	Name	1	
	-	-	-	-	Official	0-1	
	-	Executor	-	-	-	1-n	Содержит информацию об исполнителе поручения
	-	-	Organization	-	-	1	
	-	-	-	Address	-	0-1	
	-	-	-	Econtact	-	0-n	
	-	-	-	OfficialPerson	-	0-1	
	-	-	-	-	Name	1	
	-	-	-	-	Official	0-1	

7.1.5 Описание зоны "Расширение"

Зону сообщения "Расширение" образует элемент Expansion первого уровня иерархии и вложенные в него элементы.

Содержит информацию, не определенную в других зонах паспорта.

Наименования и уровни вложенности элементов, входящих в состав зоны сообщения "Расширение" приведены ниже (см. Таблица 8):

Таблица 8

Наименования и уровни вложенности элементов зоны «Расширение»				Кратность	Комментарий
1	2	3	4		
Expansion	-			0-1	
-	RC	-		0-n	Содержит информацию о документе, не предусмотренную в зоне «Документ»
-	File	-		0-n	Содержит описание файлов сообщения, включая файл паспорта.
-	-	Reference		0-1	Содержит ссылку на резолюцию, чей файл передается в сообщении
-	-	Confident		0-1	Содержит информацию о грифе файла документа
-	-	Restriction		0-1	Содержит информацию об ограничении доступа к файлу
-	-	EDS		0-n	Содержит файлы подписей основного файла и их описание
-	AdvFields	-		0-n	Содержит информацию о дополнительных реквизитах документа
-	Rubric	-		0-n	Содержит информацию о тематических рубриках документа
-	AdvInfo	-		0-n	Содержит дополнительную информацию об отправке документа этим сообщением
	EAddress	-		0-1	Содержит информацию об адресе, по которому необходимо отправлять уведомления о регистрации документа.
	ApplicationKind			0-1	Содержит сведения для виртуальной приемной ЦБ
	FilialCO Number			0-n	Содержит сведения о номере филиала кредитной организации (при обмене по каналу ЛК для ЦБ)
	AdvDoc			0-n	Содержит информацию о сопроводительном документе основного документа
		Organization		0-1	Содержит информацию об авторе сопроводительного документа
			Address	0-1	
			Econtact	0-n	
			Name	0-1	
		RegNumber		0-1	Содержит информацию о рег.№ и дате регистрации сопроводительного документа
	Region			0-n	Регион обжалуемых действий (при обмене с АИК Надзор)

7.1.6 Описание зоны "Уведомление"

Зону сообщения "Уведомление" образует элемент Acknowledgement первого уровня иерархии и вложенные в него элементы.

Уведомление отправляется в ответ на поступившее сообщение и содержит информацию о регистрации пришедшего документа или об отказе от регистрации.

Наименование и уровень вложенности каждого элемента, входящего в состав зоны сообщения "Уведомление" приведены ниже. (см. Таблица 9):

Таблица 9

Наименование и уровень вложенности элементов		Кратность	Комментарий
1	2		
Acknowledgement	-	0-1	Кратность = 1 для сообщений вида: "Уведомление", для всех остальных сообщений кратность = 0
-	RegNubmer	0-1	Регистрационный номер документа, присвоенный в системе-получателе. Кратность = 1 для сообщений об успешной регистрации документа (вид сообщения - "Уведомление о регистрации документа")
-	AckResult	1-n	Содержательная часть уведомления

7.1.7 Описание элементов

В настоящем разделе приводится описание элементов паспорта сообщения. Элементы приводятся в алфавитном порядке.

7.1.7.1 Элемент Acknowledgement

Назначение: содержит информацию о регистрации или отказе от регистрации присланного документа.

Содержание: нет.

Атрибуты: допустимые атрибуты приведены ниже. (см. Таблица 10):

Таблица 10

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
msg_id	1	String	Уникальный служебный идентификационный номер поступившего сообщения	= "Header / msg_id" инициативного сообщения о пересылке документа
ack_type	1	Enum	Вид уведомления	= 3 для уведомления о регистрации документа в СЭД получателя; = 4 для уведомления о об отказе в регистрации = 5 для уведомления о полной досылке реквизитов

7.1.7.2 Элемент AckResult

Назначение: содержательная часть уведомления о регистрации или отказе в регистрации.

Содержание: нет.

Атрибуты: допустимые атрибуты приведены ниже. (см. Таблица 11):

Таблица 11

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
errorcode	1	Num	Код ошибки	= 0 по умолчанию
errortext	0	String	Описание причины отказа от регистрации	

7.1.7.3 Элемент Address

Назначение: описание почтового адреса.

Содержание: нет.

Атрибуты: допустимые атрибуты приведены ниже. (см. Таблица 12):

Таблица 12

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
settlement	0	String	Название населенного пункта (города, поселка и т.п.)	<= 255 байт
region	0	String	Название региона (республики, края, области, автономного округа, автономной области)	<= 64 байт
postcode	0	String	Почтовый индекс	<= 12 байт
nontypical	0	String	Прочие элементы почтового адреса (улица, номер дома и т.п.)	<= 255 байт

7.1.7.4 Элемент Addressee

Назначение: описание адресата документа.

То, кому адресуется документ - организации (юридическому лицу) или гражданину (физическому лицу) - определяется соответствующим вложенным элементом – Organization или PrivatePerson.

Содержание: нет.

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 13):

Таблица 13

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
type	1	Enum	Тип адресата	= «ю», если адресат – юридическое лицо; = «ф», если адресат физическое лицо;
answer_id	0-1	String	Уникальный служебный идентификационный номер адресата	Для сообщения с документом значение этого атрибута возвращается отправителю сообщения в сообщениях – уведомлениях для связывания последних с адресатом инициативного сообщения.

7.1.7.5 Элемент Author

Назначение: описание автора документа.

Для элемента Author тот, кто является автором документа – организация (юридическое лицо) или гражданин (физическое лицо) - определяется соответствующим вложенным элементом – Organization или PrivatePerson.

Содержание: нет

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 14):

Таблица 14

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
type	1	Enum	Тип адресата	= "ю", если автор – юридическое лицо; = "ф", если автор физическое лицо;

7.1.7.6 Элемент AdvDoc

Назначение: Содержит информацию о сопроводительном документе основного документа.

Содержание: Нет

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 15).

Таблица 15

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
annotation	0	String	Содержание сопроводительного документа	<= 2000 байт
consists	0	String	Состав документа	<= 64 байт
note	0	String	Доп. информация о сопроводительном документе	<= 2000 байт

7.1.7.7 Элемент AdvFields

Назначение: содержит информацию о дополнительных реквизитах РК.

Содержание: наименование дополнительного реквизита

Тип данных: String. (<= 64 байт)

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 16):

Таблица 16

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
type	1	Enum	Тип реквизита	Допустимые значения: text / date / decimal / flag
sys_name	0	String	Уникальное системное имя реквизита в системе отправителя	<= 24 байт
data	0	String	Значение реквизита	Для реквизитов типа "текст" <= 255 байт

7.1.7.8 Элемент AdvInfo

Назначение: дополнительная информация о пересылке документа.

Содержание: текст дополнительной информации

Тип данных: String. (<= 2000 байт)

Атрибуты: нет.

7.1.7.9 Элемент **Confident**

Назначение: характеристика ограничений доступа к документу (гриф документа) или поручению.

Содержание: название ограничения (грифа), согласно принятым в организации регламентирующим документам.

Тип данных: String. (<= 64 байт)

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 17):

Таблица 17

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
flag	1	Enum	Признак ограничения доступа к документу	= 0 для открытого документа; = 1 для документа с ограниченным доступом

7.1.7.10 Элемент **document**

Назначение: основные реквизиты пересылаемого документа.

Содержание: нет.

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 18):

Таблица 18

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
idnumber	1	String	Уникальный служебный идентификационный номер документа в системе отправителя.	<= 10 байт
type	1	Enum	Тип документа в системе отправителя	= 0 для исходящего документа; = 1 для входящего документа (письма гражданина); = 2 для проекта документа (используется в функции выгрузки проекта документа)
kind	0	String	Вид документа (группа документа)	
pages	0	Num	Количество листов документа	
annotation	0	String	Краткое содержание документа	(<= 2000 байт)
collection	0	Enum	Признак коллективности обращения гражданина	= 0 для обращения, не являющегося коллективным; = 1 для коллективного обращения
UID	0	String	UID документа в передающей системе	Обязательно заполняемый атрибут для основного пересылаемого документа

7.1.7.11 Элемент EAddress

Назначение: адрес электронной почты; данный адрес используется для отправки квитанции о регистрации или отказе от нее

Содержание: значение адреса

Тип данных: String. (<= 255 байт)

Атрибуты: нет.

7.1.7.12 Элемент Econtact

Назначение: номера (адреса) имеющихся средств электросвязи.

Содержание: допускается последовательное перечисление через запятую нескольких номеров (адресов).

Тип данных: String⁵.

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 19):

Таблица 19

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
type	0	Enum	Тип номера (адреса)	= "п" для адреса эл. почты (email); = "д" для домашнего телефона;

7.1.7.13 Элемент EDS

Назначение: файлы отсоединённых подписей и их описание.

Содержание: двоичное содержимое файла подписи, преобразованное в hex-строку

Тип данных: String.

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 20):

Таблица 20

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
date	1	DateTime	Дата и время подписания	Используется время UTC
id_kind	0	Enum	Код вида подписи	Возможны следующие значения: 0; 1; 2; 3; 4; 5.
kind	0	String	Наименование вида подписи	Возможны следующие значения: = "Не определен"; = "Авторская"; = "Согласующая"; = "Утверждающая"; = "Удостоверяющая"; = "Ознакомительная". Значение этого атрибута должно соответствовать значению атрибута id_kind.
certificate_owner	0	String	Информация о владельце сертификата	<= 2000 байт.
certificate	0	String	Информация об идентификаторе сертификата	<= 2000 байт. Элемент используется для совместимости с предыдущими версиями

⁵ Для элемента с типом "п" максимальная длина содержания не должна превышать 255 байт, а для элемента с типом "д" - 24 байта.

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
				паспорта

7.1.7.14 Элемент Executor

Назначение: информация об исполнителе поручения по документу.

Содержание: нет.

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 21):

Таблица 21

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
responsible	1	Enum	Метка ответственного исполнителя	= 0 для соисполнителя; = 1 для ответственного исполнителя
deadline	0	Date	Срок исполнения	Срок исполнения задания для конкретного исполнителя

7.1.7.15 Элемент Expansion

Назначение: определение стандартного пути расширения стандарта.

Содержание: нет.

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 22):

Таблица 22

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
organization	1	String	Наименование организации – разработчика расширения	
exp_ver	1	String	Версия зоны "Расширение"	

7.1.7.16 Элемент Figurant

Назначение: сведения о фигуранте.

Содержание: нет.

Атрибуты: нет.

7.1.7.17 Элемент File

Назначение: информация о пересылаемых в сообщении файлах

Содержание: нет.

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 23):

Таблица 23

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
location	1	String	Способ передачи файла	= «out»
name	1	String	Имя файла, с которым он передается в сообщении	Имя должно быть уникально среди файлов сообщения. Имя файла паспорта = «passport.xml» <= 255 байт.
description	1	String	Наименование файла в системе отправителя	= «Паспорт РК» для файла с именем «passport.xml»; <= 255 байт
type	1	Enum	Тип пересылаемого	= 0, если это файл Паспорта,

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
			файла	= 1, если это файл пересылаемой РК; = 2, если это файл Поручения (с ЭП резолюции) = 3, если это файл Поручения
UID	0	String	UID файла в передающей системе	Обязательно заполняемый атрибут для пересылаемого файла (только для файлов, прикрепленных к РК или поручениям, а также для файла подписи резолюции)
ETag	0	String	Уникальная электронная метка файла в передающей системе	Обязательно заполняемый атрибут для пересылаемого файла (только для файлов, прикрепленных к РК или поручениям, а также для файла подписи резолюции)

7.1.7.18 Элемент FilialCONumber

Назначение: номер филиала кредитной организации.

Содержание: указывается номер филиала кредитной организации.

Тип данных: String (<= 255 байт).

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 25):

Таблица 24

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
shortname	1	String	Краткое наименование головной кредитной организации	= значению атрибута shortname элемента Organization, номер филиала которой указан в содержании данного элемента

7.1.7.19 Элемент header

Назначение: корневой элемент паспорта - заголовок сообщения, общее описание сообщения.

Содержание: нет.

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 25):

Таблица 25

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
standart	1	String	Вид стандарта, по которому создано данное сообщение	= «Стандарт ДОУ»
version	1	String	Версия стандарта	= «1.0»
time	1	Date, DateTime	Дата формирования сообщения	=TODAY Для всех типов документов, кроме refusal и receipt, для которых должен использоваться тип данных

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
				DateTime
msg_type	1	Enum	Вид сообщения Влияет на перечень допустимых элементов (зон) в сообщении	Значение = 0 для уведомления; Значение =1 для основного документа;
msg_id	0	String	Уникальный служебный идентификационный номер сообщения	Для сообщения о пересылке документа значение этого атрибута возвращается отправителю сообщения в сообщениях – уведомлениях для связывания последних с инициативным сообщением. Для уведомления о регистрации = уникальному служебному идентификационному номеру зарегистрированного документа. = 0, для уведомления об отказе от регистрации
msg_acknow	0	Enum	Подписка на уведомления	= 0 при отсутствии необходимости посылки уведомлений; Значение = 2 при необходимости посылки уведомлений.
from_org_id	1	String	Уникальный служебный идентификационный номер организации - отправителя	
from_organization	1	String	Наименование организации -отправителя	
from_department	0	String	Наименование подразделения -отправителя	
from_sys_id	1	String	Уникальный служебный идентификационный номер системы отправителя	
from_system	1	String	Наименование системы управления документами отправителя	
from_system_details	0	String	Дополнительные данные о системе управления документами отправителя	Номер версии системы
to_org_id	0	String	Уникальный служебный идентификационный номер получателя	При отправке уведомлений и ответных сообщений об исполнении ранее направленного документа, значения атрибутов рекомендуется брать из атрибутов from_... принятого соответствующего сообщения
to_organization	0	String	Организация- получатель	
to_department	0	String	Подразделение- получатель	
to_sys_id	0	String	Уникальный служебный идентификационный номер системы получателя	

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
to_system	0	String	Наименование системы управления документами получателя	
to_system_details	0	String	Дополнительные данные о системе управления документами получателя	
time2	1	DateTime	Дата и время формирования сообщения	= NOW

7.1.7.20 Элемент Name

Назначение: фамилия, имя, отчество (ФИО).

Содержание: ФИО.

Тип данных: String. (<= 64 байта)

Атрибуты: нет.

7.1.7.21 Элемент Official

Назначение: описание штатной единицы (подразделение, должность), занимаемой должностным лицом.

Содержание: нет.

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 26):

Таблица 26

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
department	0	String	Наименование подразделения	<= 255 байт
post	0	String	Наименование должности	<= 255 байт

7.1.7.22 Элемент OfficialPerson

Назначение: контейнер для описания должностного лица.

Содержание: нет.

Атрибуты: нет.

7.1.7.23 Элемент Organization

Назначение: описание организации.

Содержание: нет.

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 27):

Таблица 27

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
fullname	0	String	Полное название организации	<= 255 байт
shortname	1	String	Краткое название организации	<= 255 байт
ogrn	0	String	Основной государственный регистрационный номер	<= 64 байт
inn	0	String	Идентификационный номер налогоплательщика (ИНН)	<= 64 байт

7.1.7.24 Элемент OutNumber

Назначение: контейнер для регистрационного номера документа.

Содержание: нет.

Атрибуты: нет.

7.1.7.25 Элемент PrivatePerson

Назначение: описание физического лица.

Содержание: нет.

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 28):

Таблица 28

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
inn	0	String	Индивидуальный номер налогоплательщика	<= 64 байт
pas_ser	0	String	Серия паспорта	
pas_num	0	String	Номер паспорта	
pas_org	0	String	Кем и когда выдан паспорт	<= 255 байт

7.1.7.26 Элемент RC

Назначение: реквизиты РК, не вошедшие в зону "Документ".

Содержание: нет.

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 29):

Таблица 29

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
rc_comment	0	String	Примечание к РК документа	<= 2000 байт
adr_comment	0	String	Примечание к адресату документа	<= 2000 байт
consists	0	String	Состав документа, направляемого адресату	<= 255 байт

7.1.7.27 Элемент Reference

Назначение: ссылка на резолюцию к документу, чей файл с подписью пересылаем.

Содержание: Уникальный служебный идентификационный номер, пересылаемого поручения

Тип данных: String.

Атрибуты: нет.

7.1.7.28 Элемент Referred

Назначение: ссылка на документ или поручение, к которым относится поручение.

Содержание: нет.

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 30):

Таблица 30

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
idnumber	0	String	Уникальный служебный идентификационный номер документа или родительского поручения	Ссылка на родительское поручение дается только в том случае, если оно тоже пересылается в этом сообщении.

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
retype	0	String	Определяет вид ссылки	= "д", если передаем ссылку на документ; = "з", если передаем ссылку на поручение.

7.1.7.29 Элемент Region

Назначение: Регион обжалуемых действий.

Содержание: наименование региона обжалуемых действий.

Тип данных: String. (<= 2000 байт)

Атрибуты: нет.

7.1.7.30 Элемент RegNumber (RegNumer)

Назначение: регистрационный номер и дата регистрации документа. Для документов, поступивших от физических лиц, допускается указание только даты.

Содержание: регистрационный номер документа.

Тип данных: String. (<= 64 байт)

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 31):

Таблица 31

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
regnum	0	String	Регистрационный номер	<= 64 байта
regdate	0	Date	Дата регистрации	

7.1.7.31 Элемент Restriction

Назначение: ограничение уровня доступа к файлу

Содержание: текстовое описание ограничения

Тип данных: String. (<= 64 байт)

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 32):

Таблица 32

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
data	1	Enum	Код уровня доступа	1 - «Картотечный/Простой», 2 - «Фигуранты и список ДЛ», 3 - «Список ДЛ», 4 - «Строгий - Фигуранты и список ДЛ», 5 - «Строгий - Список ДЛ».

7.1.7.32 Элемент Rubric

Назначение: тематическая рубрика документа.

Содержание: наименование рубрики.

Тип данных: String. (<= 2000 байт)

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 33):

Таблица 33

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
code	0	String	Код рубрики	<= 64 байта

7.1.7.33 Элемент SignDate

Назначение: указание даты подписания.

Содержание: дата подписания документа.

Тип данных: Date.

Атрибуты: нет.

7.1.7.34 Элемент SNILS

Назначение: СНИЛС гражданина.

Содержание: Значение СНИЛС гражданина.

Тип данных: String. (<= 14 символов)

Атрибуты: нет.

7.1.7.35 Элемент Task

Назначение: содержит информацию о поручении, пересылаемом вместе с документом.

Содержание: нет.

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 34):

Таблица 34

Имя атрибута	Кратность	Тип данных	Описание	Правила заполнения
idnumber	1	String	Уникальный служебный идентификационный номер поручения в СЭД отправителя	
task_reg	1	Enum	Отметка о регистрации поручения в СЭД отправителя	= 1
task_copy	1	Enum	Отметка о передаче копии задания	= 0
kind	0	String	Вид поручения	Допустимые значения: «пункт» / «резолуция»;
task_text	0	String	Текст поручения	<= 2000 байт
deadline	0	Date	Срок исполнения поручения	
UID	0	String	UID поручения в передающей системе	Обязательно заполняемый атрибут для пересылаемого поручения

7.1.7.36 Элемент TaskNumber

Назначение: номер пункта документа или дата резолюции по документу.

Содержание: номер поручения (пунктов).

Тип данных: String (<= 64 байт)

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 35):

Таблица 35

Имя допустимого атрибута	Кратность	Тип данных	Описание	Правила заполнения
taskDate	0	Date	Дата резолюции	

7.1.7.37 Элемент Validator

Назначение: Информация о согласовании (визировании) документа.

Содержание: нет.

Атрибуты: допустимые атрибуты приведены ниже (см. Таблица 36):
Таблица 36

Имя допустимые атрибута	Кратность	Тип данных	Содержание	Правила заполнения
type	1	String	Тип визирующего	= "ю", юридическое лицо
attestation	1	String	Гриф согласования или утверждения	= "Согласовано";

7.1.7.38 Элемент Writer

Назначение: исполнитель (составитель) документа.

Содержание: = "ю"

Тип данных: String

Атрибуты: нет.

8 Дополнительные функции (для ДЕЛО API 2009 и ДЕЛО API 2023)

8.1 API входа в систему «ДЕЛО» и выхода из системы

API доступны по адресу /CoreHost/identity/api/

Создание сессии выполняется путем вызова POST запроса <адрес сервиса>/CoreHost/identity/api/login с параметрами **user** и **password**, а также передать элемент **app** в коллекции **items**. Клиент должен поддерживать корректную обработку cookies.

В случае успешного первичного входа также могут быть запрошены дополнительные проверки, например EMail, запрос на удаление существующей сессии пользователя. Если требуются дополнительные проверки, сервер возвращает ответ 400 со специальным объектом ответа, где указывается, какая ошибка возникла при аутентификации.

В случае успешного входа создается постоянная Cookie, в случае ошибки возвращается объект с текстом ошибки.

Заккрытие сессии выполняется путем вызова POST запроса: <адрес сервиса>/CoreHost/identity/api/logout

Пример входа и выхода:

Запрос:

```
Invoke-WebRequest <адрес сервиса>/CoreHost/identity/api/login -SessionVariable session -Method Post -ContentType "application/json" -Body (@{user = "tver"; password = "tver"; items = @{ app = "test" }} | ConvertTo-Json) | % Content
```

Вывод:

```
{"NeedRedirect":true,"Url":"/ArmSite/Site/ArmMain.html","Code":null,"Message":null}
```

Запрос:

```
Invoke-WebRequest <адрес сервиса>/CoreHost/identity/api/logout -WebSession $Session -Method Post | % Content
```

Вывод:

```
{"NeedRedirect":false,"Url":null,"Code":null,"Message":null}
```

В сессиях пользователя нужно отличать "обычные" UI-сессии и интеграционные, которые открыты через API (пользовательские приложения, фоновые задачи).

app – это специальный параметр, который обязателен. Он является признаком интеграционной сессии, под определенным логином и паролем. В режиме app не удаляются старые сессии и отсутствуют проверки на дубли сессий и проверки лицензий.

8.2 API пользовательских настроек (USER_SETTINGS)

Список доступных API (см. Таблица 37):

Таблица 37

Описание	Параметры	Метод	Возвращает	Описание
user/usersettings?category={0}×tamp={1}&systemsettings={2}&module={3}	category, timestamp, systemsettings, module (категория, отметка времени, системные настройки, модуль)	Get	File(json)	Метод для получения json по category и временной отметке. Если systemsettings=true, настройки пишутся с идентификатором -99. (по умолчанию - false) module имеет значение по умолчанию 'none'

user/usersettings/upload?iswholejson={0}	iswholejson (полностью заменить json)	Post		Метод для загрузки файла json в систему. В тело запроса передается файл json или поток данных.
--	--	------	--	--

8.3 API системных настроек (APP_SETTINGS)

Адрес контроллера: <адрес сервиса>/CoreHost/appsettings (см. Таблица 38):

Таблица 38

Описание	Параметры	Метод	Возвращает	Описание
{name}?namespace={0}&typename={1}&instance={2}&mergedefault={3}	namespace, typename, instance (по умолчанию - "Default") (все string), mergedefault (по умолчанию - true, bool) (Пространство имен, тип, экземпляр)	Get	File (json)	Метод для получения определенного класса в виде json. Если параметр mergedefault - true, метод get возвращает полную копию объекта instance, но если в этом объекте какие-то поля не заполнены, они будут по умолчанию взяты из объекта instance="Default"
{name}/upload?namespace={0}&typename={1}&instance={2}	namespace, typename, instance (по умолчанию - "Default") (все string) (Пространство имен, тип, экземпляр) Передача класса в виде json как StreamContent в Body	Post		Метод для загрузки класса в систему.
{name}/get-list?namespace={0}&typename={1}&mergedefault={2}	namespace, typename, (все string), mergedefault (по умолчанию - true, bool) (Пространство имен, тип)	Get	File (json)	Метод для получения словаря объектов определенного типа в виде json (ключ словаря - имя экземпляра). Если параметр mergedefault - true, метод get-list возвращает полную копию объектов словаря, но если в любом объекте списка какие-то поля не заполнены, они будут по умолчанию взяты из объекта instance="Default"

Запрос:

```
Invoke-WebRequest -Uri <адрес сервиса>
/CoreHost/appsettings?namespace=Eos.Delo.Settings.Common&typename=FilesCfg" -
WebSession $session | % Content
```

Вывод:

```
{"Library":{"Name":"Storage","Directory":"FileStorage"},"EdmsParm":"STORAGE","
MaxFileSize":500}
```

8.4 Работа с временным хранилищем файлов FDULZ

8.4.1 Помещение файла во FDULZ

Выполняется путем вызова POST запроса <адрес сервиса>/CoreHost/OData/fdulz/CoreHost/OData/fdulz/UploadFile с отправкой формы в теле имени файла и содержимого:

```
$FileName = Split-Path $FilePath -Leaf
$Response = Invoke-WebRequest -Uri <адрес
сервиса>/CoreHost/fdulz/api/UploadFile -Method 'POST' -WebSession $Session `
-ContentType "application/x-www-form-urlencoded; charset=utf-8" `
-Form @{inputFileName = $FileName;file = Get-Item -Path $FilePath}
$fdulzid = $Response.Content
Write-Host "fdulzid: ", $fdulzid
```

8.4.2 Использование содержимого файла из FDULZ в OData

Производится запуском в составе пакетной команды сервисной операции REF_FILE_SetFdulzContents?sf=<имя файла во FDULZ>&tf=<идентификатор файла>.

Например:

```
[{
"method": "POST",
"url": "PRJ_RC(4701)/REF_FILE_List",
"data": {
"SEND_ENABLED": 1,
"ISN_FILELINK": 5,
"DESCRIPTION": "c.txt",
"KIND_DOC": 7,
"DONTDEL_FLAG": 0,
"ISN_REF_FILE": -19999,
"ISN_REF_DOC": 4701,
"SECURLEVEL": 1,
"ACCESS_MODE": 1,
"APPLY_EDS": 1,
"IS_HIDDEN": 0,
"LOCK_FLAG": 0,
"ORDERNUM": 1,
"EDS_CNT": 0
}
},
{
"method": "POST",
"url": "REF_FILE_SetFdulzContents?tf=-
19999&sf='d01dd9930a044faea9df0dede39cbfd2.txt'"
}
```

8.4.3 Использование содержимого файла из FDULZ в GraphQL

Для того, чтобы прикрепить файл к объекту нужно отправить соответствующий GraphQL запрос на gql/query:

```
# Прикрепление файла к объекту
$text = 'mutation {
  createTempRc(input:{
    clientMutationId:"idl"
    data:
    {
      refFiles:[
```

```

        {
            createRefFile: {
                contents: "fdulz#DeleteOnClose#2c945409-45d2-467f-
8ee4-9f31956bbb90"
                description: "1.txt"
            }
        }
    ]
}
}))
{
    success
    message
    messageCode
    systemMessage
    data {
        isnTempRc
    }
}
}'

$json = @{query = $text; variables = $null; } | ConvertTo-Json

$Response = Invoke-WebRequest -Uri "$BaseUrl/CoreHost/gql/query" -Body $json -
Method 'POST' -WebSession $Session -ContentType "application/json;
charset=utf-8"

```

В поле типа Handle (в GraphQL схеме это строка) нужно поместить строку формата `fdulz#DeleteOnClose#FdulzId`, где:

- `fdulz` - это обязательный префикс;
- `#` - разделитель между параметрами;
- `DeleteOnClose` - необязательный параметр, который определяет действие с файлом после завершения загрузки в систему, а именно удаление;
- `FdulzId` - обязательный параметр, идентификатор файла во FDULZ хранилище, например, может являться типом `guid`.

Второй параметр необязательный, т.е. можно отправить строку вида `fdulz#FdulzId` или `fdulz##FdulzId`, но в таком случае удаление файла из FDULZ хранилища выполняться не будет.

8.4.4 Чтение файла из FDULZ

Чтение файла из FDULZ можно осуществить, имея значение идентификатора файла ранее помещенного во FDULZ. Для этого нужно вызвать метод `DownloadFile` передав в него идентификатор файла. После чтения файл удаляется из хранилища FDULZ, если не передать в строке запроса параметр `nodelete`.

Пример:

```

$Response = Invoke-WebRequest -Uri "<адрес
сервиса>/CoreHost/fdulz/api/DownloadFile?fileId=$fdulzid&nodelete" -Method
'GET' -WebSession $Session

```


9 Разработка расширений REST API (для ДЕЛО API 2023)

9.1 Пользовательские запросы

Пользовательские запросы позволяют создавать на сервере приложений любую специальную логику и впоследствии вызывать ее через REST API GraphQL.

Для пользовательского запроса необходимо определить атрибут метода подобного вида:

```
[Api(RequestType.Query, FieldName = "getCurrentUserCl", Description = "Возвращает текущего пользователя.")]
```

Где:

- Api - определяет, что это пользовательский запрос,
- RequestType - тип запроса (Query - запрос на чтение, Mutation - запрос на изменение),
- FieldName - название функции в GraphQL,
- Description - описание, видимое при просмотре схемы GraphQL

9.1.1 Запросы на чтение

Пользовательские запросы на чтение позволяют либо возвращать значение (строка, число, объект, список и т.д.), либо добавлять к существующей таблице дополнительные поля.

Пример запроса:

```
// Запрос на чтение - возвращает текущего пользователя
[Api(RequestType.Query, FieldName = "getCurrentUserCl", Description =
"Возвращает текущего пользователя.")]
public async Task<IUserCl> GetCurrentUserClAsync(MiddlewareContextBase
baseContext)
{
    var currentUser = baseContext.GetDeloContext().User;

    await _extProtHelper.WriteUserViewAsync(currentUser.UserCl,
baseContext.CancellationToken);

    return currentUser.UserCl;
}
```

Пример запроса к существующей таблице:

```
// Запрос на чтение к существующей таблице - возвращает список прав доступа
[Api(RequestType.Query, FieldName = "acl", Description = "Список прав
доступа")]
public async Task<IDictionary<IWapiSession, IWapiSessionAcl>>
CanTech(IEnumerable<IWapiSession> sources, MiddlewareContextBase baseContext)
{
    var currentUser = baseContext.GetDeloContext().User;

    return await currentUser.CanTech(sources, baseContext);
}
```

Пример пользовательского запроса и ответа в GraphQL:

```
# Запрос текущего пользователя
$text = '{
  getCurrentUserCl
  {
    classifName
  }
}'
```

Ответ:

```
{
  "data": {
    "getCurrentUserCl": {
```

```

        "classifName": "TVER"
    }
}
}

    Ответ:
{
  "data": {
    "getCurrentUserCl": {
      "classifName": "TVER"
    }
  }
}

```

9.1.2 Запросы на изменение

Запросы на изменение данных работают как обычная мутация:

```

// Запрос на изменение - блокирует заданные файлы
[Api(RequestType.Mutation, FieldName = "sampleLockRefFiles", Description =
"Блокирует заданные файлы.")]
    public async Task SampleLockRefFilesAsync(LongIdsArgument input,
MiddlewareContextBase baseContext) =>
        await _mutationHelperService.LockRefFilesAsync(input.Ids,
baseContext);

```

Вызов подобного запроса в GraphQL будет выглядеть так:

```

# Запрос на изменение - блокирует заданные файлы
$text = 'mutation {
  sampleLockRefFiles(input:
  {
    clientMutationId: "id2",
    ids:[502]
  })
  {
    success
    message
    messageCode
  }
}'

```

9.2 Контроллеры

Пользователь может создавать собственные контроллеры и обращаться к методам этих контроллеров через REST API.

Приведем пример контроллера и рассмотрим его:

```

// Пример контроллера
using Eos.Delo.Platform.Storage.Constants;
using Eos.Delo.Platform.Storage.Helpers;
using Eos.Delo.Platform.Storage.Model;
using Eos.Delo.Platform.Storage.Services.Caching;
using Eos.Delo.Samples.ControllersAndCustomQueries.Custom;
using Microsoft.AspNetCore.Mvc;
using System.Text.Json;
using System.Text.Json.Serialization;
using System.Threading;
using System.Threading.Tasks;

namespace
Eos.Delo.Samples.ControllersAndCustomQueries.Areas.Sample.Controllers
{
    [Authentication]
    [Route("sample/samplesample")]
    public class SimpleSampleController : ControllerBase
    {
        private readonly UserCachingService _userService;
    }
}

```

```

private readonly CustomQueriesSample _customQueries;

public SimpleSampleController(UserCachingService userService,
CustomQueriesSample customQueries)
{
    _userService = userService;
    _customQueries = customQueries;
}

[HttpGet("getdepartmentwithparent")]
public async Task<ActionResult>
GetDepartmentWithParentAsync([FromServices] IDataContext db, [FromQuery(Name =
"isnnode")] long isnNode)
{
    var cancellation_token = new CancellationToken();

    var user = await _userService.GetUserAsync(User,
cancellation_token);

    var baseContext = MiddlewareHelper.CreateQueryContext(db, user,
null, ExtProtOperationType.None, cancellation_token);

    var result = await
_customQueries.GetDepartmentWithParentAsync(isnNode, baseContext);

    return Ok(result);
}

[HttpPost("createtestdepartment")]
public async Task<ActionResult>
CreateTestDepartmentAsync([FromServices] IDataContext db)
{
    var cancellation_token = new CancellationToken();

    var user = await _userService.GetUserAsync(User,
cancellation_token);

    var baseContext = MiddlewareHelper.CreateMutationContext(db, user,
null, ExtProtOperationType.Mutation, cancellation_token);

    await _customQueries.CreateTestDepartmentAsync(baseContext);

    return Ok();
}
}

```

Контроллер определяется атрибутом `Route`, где прописывается путь к контроллеру. Класс контроллера наследуется от `ControllerBase`, но также можно создавать базовые контроллеры, от которых будут наследоваться основные. Сервисы внедряются стандартно через DI в конструкторе, либо через атрибут `FromServices` прямо в методе. Затем можно определить методы `Get` и `Post` с помощью одноименных атрибутов. Для данных методов также можно прописать конкретный путь, например `[HttpGet("some-address")]`. В дальнейшем работа с методами контроллера идет стандартными средствами через HTTP.

10 Разработка отчётов

Примеры реализации с комментариями можно посмотреть в решениях-примерах: Eos.Delo.Sample.Reports.OnP и Eos.Delo.Sample.Reports.Module

10.1.1 Создание и настройка модуля отчётов

Создание модуля с отчётами, на примере Visual Studio 2022:

- 1) Создаём решение и проект на основе шаблона: Пустой шаблон ASP.NET Core (решение назовем Sample, проект назовём аналогично);
- 2) Платформа: .NET 6;
- 3) Настроить для HTTPS – да.

Подключаем NUGET пакет: Eos.Delo.Srch, версия 24.2.* - где * последняя доступная версия пакета по возрастанию.

После создания проекта:

- 1) Нажмём правой кнопкой мыши по нашему проекту внутри решения – Свойства – Общие – Установим тип вывода на – Библиотека классов.
- 2) Удаляем файл appsettings.json и Program.cs.
- 3) Создаём файл Startup.cs.
- 4) В корне решения создадим файл XML файл с названием building.xml.
- 5) Переименуем файл в building.props.
- 6) Внутри файла пропишем следующую структуру.

```
<Project xmlns="http://schemas.microsoft.com/developer/msbuild/2003">
  <PropertyGroup>
    <EosSrchVersion>24.2.*</EosSrchVersion>
  </PropertyGroup>
</Project>
```

В данном файле мы настраиваем автоматическое использование пакета Srch 24.2.* последней доступной версии.

Перейдём в корневой файл нашего проекта - Sample.csproj и внесём некоторые изменения.

Подключим использование созданного файла с расширением props

```
<Import Project="..\building.props" />
```

Настроим использование пакета Eos.Delo.Srch следующим образом:

```
<ItemGroup>
  <PackageReference Include="Eos.Delo.Srch" Version="$(EosSrchVersion)"
ExcludeAssets="runtime" />
</ItemGroup>
```

Теперь в нашем проекте будет использоваться последняя актуальная версия пакета, прописанная в файле с расширением props.

Данный метод хорошо подходит, если у нас в решении, например, лежит 2 и более отчёта, нам не нужно руками в каждом обновлять пакет, а делаем это, в 1 файле.

Перейдём в ранее созданный класс Startup.cs нашего проекта и сделаем его модулем, для этого, наследуемся от интерфейса IModule из Eos.Platform.Hosting.Module.

```
using Eos.Delo.Abstractions.Srch;
using Eos.Platform.Hosting.Module;
```

```
public class Startup : IModule
```

И реализуем его.

```
public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
{}
```

```
public void ConfigureServices(IServiceCollection services)
{}
```

С созданием и первоначальными настройками закончено, приступим к созданию отчёта.

10.1.2 Создание отчёта

Создаём Razor файл для отчёта, например Report10.cshtml, в котором реализуем поиск документов, зарегистрированных в выбранной картотеке и которые относятся к выбранной группе документов.

Удаляем всё содержимое из созданного файла.

Устанавливаем пространство имён: @namespace wrep - обязательно wrep

Подключаем ссылку: @using Eos.Delo.Srch.Reports

Наследуемся от класса ProtoRep который находится в пакете Eos.Delo.Srch и от двух дополнительных интерфейсов ICriteriesForm и IMount.

Реализуем класс, создаём блок функций:

```
@functions
{
  ..
}
```

Внутри данного блока создаём список UOD критериев, которые будут отображаться на визуальной части приложения (фронт):

```
Uod[] _uodCriteries = new[]
{
  Uod("regcard", "Картотека", "uod_classif", "source(CARDINDEX) return_due(true) rb_
data(;Текущая картотека;) val(; $CURRENT_CARD;) required(true)"),
  Uod("docgroup", "Группа документов", "uod_classif",
"source(DOCGROUP_CL) is_node(true) required(true)"),
  Uod("regdate", "Дата пер. ПК", "un_period_date_srch", "required(true)", $"{new
DateTime(DateTime.Now.Year, 1, 1).ToString("dd/MM/yyyy")}: {new
DateTime(DateTime.Now.Year, 12, 31).ToString("dd/MM/yyyy")}", "date.range")
};
```

Список UOD контроллов их описание и взаимодействие находится в PDF файле внутри примеров.

В source(...) указывается имя таблицы из БД.

Реализуем интерфейс ICriteriesForm который необходим для получения UOD-ов для отображения.

```
Uod[] ICriteriesForm.GetUods() => _uodCriteries;
```

Реализуем интерфейс IMount, который отвечает за регистрацию отчётов в системе:

```
void IMount.AutoReg()
```

```
{
    AutoReg("Сведения о документообороте", 2, 1);
}
```

Где:

- 1) Имя отчёта для отображения на фронте приложения;
- 2) Дополнительная группа для отчёта, если указана иная группа от 1, отчёт попадает в дополнительную группу и группу 1, в нашем случае – 2;
- 3) Версия отчёта. при внесении каких-либо изменений в Название отчёта, или в группу критериев UOD нужно изменять "версия" на +1.

Создадим модель данных для отчёта.

```
public class Row // основная модель
{
    public string Due; // ID группы
    public string Name; // название
    public int Node; // "родитель", добавлен для сборки диаграммы, можно
    собирать и по Layer, там будет 3 вариации, тут же их 2.
    public int? Layer; // слой
    public int[] Value; // колонки под заполнения

    public static Row Create(DOCGROUP_CL_Entity docgroup)
    {
        return new Row()
        {
            Due = docgroup.DUE + (docgroup.IsLeaf() ? "" : "%"),
            Name = docgroup.CLASSIF_NAME,
            Layer = docgroup.DUE.Split('.').Count() - 2,
            Node = docgroup.IS_NODE,
            Value = new int[1],
        };
    }
}
```

Инициализируем переменную для хранения результата и вывода на экран.

```
private List<Row> Result = new();
```

Реализуем основной метод для формирования отчёта.

```
public override void Fill(string method){}
```

Внутри метода начнём создавать реализацию отчёта.

Создадим переменную, которая будет содержать критерии для подстановки в параметры запроса поиска в БД.

Подключаем ссылку: @using Eos.Delo.Common

```
var args = new CmdArguments();
```

Получим дату регистрации, выбранную на фронте приложения через UOD-ы. DateCondition - умеет учитывать тип БД MSS/ORACLE/PGS, дополнительная конвертация не требуется.

```
args["regdate"] = Criteries.DateCondition("regdate");
```

Получаем ID выбранной картотеки, реализовав специальный метод.

```
args["cardisn"] =
GetIsnsFromDues<DEPARTMENT_Entity>(Criteriaes.DuesCrit("regcard"));
```

Метод `GetIsnsFromDues, new Query<T>` позволяет получить из Базы данных данные запрашиваемой таблицы `<T>`, так же, можно задать дополнительные критерии поиска, через: переменная `SA.Criteria[Имя переменной из БД] = Значение`.

Для работы с `new Query<T>` подключаем ссылку: `@using Eos.Delo96.Api.Entities`

```
public string GetIsnsFromDues<T>(string dues) where T : ClassifTree
{
    var q = new Query<T>(LoadMode.Table);
    q.SA.Criteria["DUE"] = dues;
    var lst = q.Search();
    return string.Join(",", lst.Select(x => x.ISN_LCLASSIF));
}
```

Делаем группы документов, которые будем перебирать для данных, так же используя `new Query<T>`.

```
foreach (var dgDue in Criteriaes.DuesCrit("docgroup").Split('|'))
{
    var qx = new Query<DOCGROUP_CL_Entity>(LoadMode.Table);
    qx.SA.Criteria["DUE"] = dgDue + "%";
    var docgroups = qx.Search().OrderBy(dg => dg.DUE);
    Result.AddRange(docgroups.Select(Row.Create).ToList());
}
```

Создадим дополнительный класс для получения результата SQL запросов из БД:

```
public class Record
{
    public int DOC_ISN { get; set; } // ID документов
}
```

Создадим файл, который будет содержать в себе список SQL команд для выполнения.

Для этого:

- 1) В корне проекта создадим папку Dictionary.
- 2) Внутри папки Dictionary создадим XML файл Cmd.xml.

Пропишем структуру файла:

```
<?xml version="1.0" encoding="utf-8" ?>
<commands xml:space="preserve">
<!-- Отчёт 10, регистрация документов -->
    <command id="Report10_command_1" description="Зарегистрированные">
        <arg name="docgroup" type="string" direction="inc"/>
        <arg name="regdate" type="string" direction="inc"/>
        <arg name="cardisn" type="string" direction="inc"/>
        <regex db="mss" pattern="--mss:"></regex>
        <regex db="ora" pattern="--ora:"></regex>
        <regex db="pgs" pattern="--pgs:"></regex>
        select rc.isn_doc as DOC_ISN
        from doc_rc rc
        inner join docgroup_cl cl on cl.due = rc.due_docgroup
        where rc.due_docgroup like '{docgroup}'
        --mss: and rc.doc_date {regdate}
```

```

--ora: and rc.doc_date {regdate}
--pgs: and rc.doc_date {regdate}
and rc.isn_card_reg in ({cardisn})
</command>
</commands>

```

К командам применяется соглашение о форматировании:

- 1) Id команды должно начинаться с имени модуля, в котором используется.
- 2) Перед набором команд писать комментарий к какому модулю (отчету) эти команды относятся.
- 3) Между командами одного набора не делать пустых строк.
- 4) Между командами разных наборов добавить две пустые строки.
- 5) Наборы команд отчетов сортировать по номеру отчета.
- 6) Regex используется для замены каких-либо данных с учётом спецификации БД (например, разная конкатенация строк или в 1 БД поле varchar, а в другой int)

При работе с SQL запросами и разными типами БД, конкатенацию строк можно заменить через регулярные выражения таким образом:

```

<regex db="ora" pattern="\>||</regex>
<regex db="pgs" pattern="\>||</regex>

```

Таким образом, при выполнении следующего запроса, + заменится на ||.

```

select (c1.SURNAME + org_reply1.CLASSIF_NAME) as SURNAME
from RESOLUTION res
inner join DEPARTMENT dp1 on dp1.DUE = res.DUE
inner join contact c1 on c1.isn_contact = dp1.isn_contact
inner join organiz_cl org_reply1 on org_reply1.isn_node = c1.isn_organiz
where res.ISN_RESOLUTION = 10

```

Начинаем перебирать выбранные группы и строить результат отчёта.

SQL.Read<T> - делает маппинг из SQL запроса на запрашиваемую модель:

```

foreach (var item in Result)
{
    args["docgroup"] = item.Due; // ID Группы
    var registered = Sql.Read<Record>("Sample_command_1", args); // зарег. в данной
    библиотеке

    // Подсчитываем данные с запроса для группы
    item.Value[0] = registered.Count();
}

```

Делаем визуальную отрисовку отчёта. Изначально прописываем стандартную HTML структуру документа, стили, отрисовка таблицы с результатом запроса. + перевод HTML в DOC формат

page: landscape задаёт альбомный формат, документ открывается в альбомном формате и на Windows и на Linux.

```

<!DOCTYPE html>
<!html xmlns="http://www.w3.org/1999/xhtml">
<!head runat="server">
<title>Отчёт</title>

```



```

<meta content="text/html; charset=utf-8" http-equiv="Content-Type">
<style type="text/css" media="print">
  @@page {
    size: landscape;
  }

  @@page Section1 {
    mso-page-orientation: landscape;
    margin-left: 2cm;
    margin-right: 2cm;
    margin-top: 2cm;
    margin-bottom: 1.5cm;
    mso-header-margin: 35.4pt;
    mso-header: h1;
    mso-title-page: yes;
  }

  div.Section1 {
    page: Section1;
  }

  p.MsoHeader, li.MsoHeader, div.MsoHeader {
    font-size: 10.0pt;
    font-family: 'Times New Roman';
    width: 100%;
    text-align: right;
  }
</style>
</!head>
<!body>

<div class="Section1">
  <h1>Сведения о регистрации документов</h1>
  <br />
  <table border="1">
    <thead>
      <tr class="head" style="font-weight:bold">
        <td>Группа документов</td>
        <td>Зарегистрировано</td>
      </tr>
    </thead>
    <tbody>
      @foreach (var item in Result)
      {
        <tr>
          <td>@item.Name</td>
          <td @SetDrillRedirectionAttribute("col2", item) style="text-align:center;">@item.Value[0]</td>
        </tr>
      }
    </tbody>
  </table>
</div>
</!body>
</!html>

```

Внимание! Данный блок скрипта отвечает за диаграмму, с фронта берутся данные именно из этого блока, блок ОБЯЗАТЕЛЬНО должен быть с id = showDiagram и никак иначе!

```

<script id="showDiagram" type="text/javascript">
  @Diagram()

```

</script>

Код метода @Diagram()

Подключаем ссылку: @using System.Text.Json

```
private string Diagram()
{
    /*
    * Если у нас в модели есть "родитель" то, строим диаграмму по нему,
    * т.к он содержит в себе все данные из "дочерних" элементов
    * иначе, строим по "дочерним" элементам
    */
    var selectedNode = (Result.Any(n => n.Node == 0)) ? 0 : 1;

    /*
    * Собираем объект самой диаграммы, более подробно в приложенных Sample
readme
    */

    var graph = new
    {
        title = "Работа с группами документов",
        date = $"{Criteriaes.Criteria("regdate").VALUE}",
        charts = new[]
        {
            new
            {
                title = "Зарегистрированные",
                type = new[] { "horizontalBar" },
                data = Result.Where(x => x.Node == selectedNode).Select(d => new
                {
                    name = d.Name,
                    value = d.Value[0],
                }).ToArray(),
                showCount = true,
                skipEmpty = true,
            }
        }
    };
    return JsonSerializer.Serialize(graph);
}
```

Для формирования отчётов, вместо прямых SQL запросов есть возможность использовать Entity Framework Core.

Для этого, вместо void Fill(string method), реализуем Task FillAsync(string method, IDataContext _context, IServiceScope scope)

```
public override async Task FillAsync(string method, IDataContext _context,
IServiceScope scope){}
```

Дополним критерии UOD логически удалёнными элементами.

```
Uod("log", "", "uod_checkboxgroup", "rb_data(;Включить логически
удалённые элементы справочников)val(;ld;)")
```

После этого, поднимем версию отчёта на 1 вверх.

```
AutoReg("Сведения о документообороте", 2, 2);
```

Как и в методе `void Fill()`, получим группы документов, уже используя EF Core.

```
var docgroups = await _context.DocgroupCls
    .In(groups, key => p => EF.Functions.Like(p.Due, key + "%"))
    .ToListAsync();
```

Добавим их в нашу модель.

```
Result.AddRange(docgroups.Select(Row.Create).ToList());
```

Перебираем модель, получаем зарегистрированные документы. Поиски документов реализованы через `IQueryable` для меньшей нагрузки БД.

В данном примере для большей наглядности работы с EF Core даты заданы по умолчанию при построении запроса.

```
foreach (var item in Result)
{
    var registeredQueryBuilder = _context.DocRcs.Include(d =>
d.DocgroupCl)
        .Where(x => x.IsnCardReg == 0L)
        .Where(x => x.DocDate >= new DateTimeOffset(new DateTime(2023, 1, 1),
        TimeSpan.Zero) && x.DocDate <= new DateTimeOffset(new DateTime(2024, 1, 1, 23, 59,
        59), TimeSpan.Zero))
        .Where(x => EF.Functions.Like(x.DocgroupCl.Due, item.Due + "%"));
    //дополняем его лог. удалёнными
    registeredQueryBuilder = DeletedFilter(registeredQueryBuilder,
enableDeleted);
    //IQueryable срабатывает тогда, когда есть .To.....
    var registered = await registeredQueryBuilder.ToListAsync();
    item.Value[0] = registered.Count();
}
```

Реализация метода включения логически удалённых элементов справочника (записи БД):

```
private IQueryable<IDocRc> DeletedFilter(IQueryable<IDocRc> query, bool
deleted)
{
    if (deleted)
    {
        return ApplyDeletedFilter(query);
    }
    else
    {
        return ApplyNonDeletedFilter(query);
    }
}

private IQueryable<IDocRc> ApplyDeletedFilter(IQueryable<IDocRc> query)
{
    return query.Where(x => x.DocgroupCl.Deleted == 0L || x.DocgroupCl.Deleted
== 1L);
}

private IQueryable<IDocRc> ApplyNonDeletedFilter(IQueryable<IDocRc> query)
{
    return query.Where(x => x.DocgroupCl.Deleted == 0L);
}
```

Отрисовка HTML контента остаётся такой же.

Если нам потребовалось вывести результат формирования отчёта в файл, в method приходит tofile.

В конце метода Fill() или FillAsync() после получения всех данных для дальнейшей отрисовки отчёта добавим конструкцию:

```
if (method == "tofile")
{
    ResultHeaders = new FileAnswer("Report_10.doc").ResultHeaders;
}
```

Тогда на основе HTML контента сформируется DOC файл и скачается автоматически.

10.1.3 Регистрация отчёта и команд

После того, как мы реализовали весь необходимый функционал, требуется зарегистрировать отчёт и команды (при их использовании) в Startup.cs, для этого вернёмся в данный класс и изменим методы.

Регистрируем в памяти приложения наш отчёт:

```
public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
{
    SrchRegistrator.Register(new Report10());
}
```

Каждый следующий отчёт регистрируется аналогично.

```
public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
{
    SrchRegistrator.Register(new Report10());
    SrchRegistrator.Register(new Report11());
    .....
}
```

Регистрируем используемый Cmd.xml файл указав путь одним из форматов

```
public void ConfigureServices(IServiceCollection services)
{
    CmdDescManager.LoadFragmentRes(typeof(Startup).Assembly,
    "Dictionary.Cmd.xml");
}
```

Или

```
CmdDescManager.LoadFragmentRes("Report10.Dictionary.Cmd.xml, Report10");
}
```

10.1.4 Переадресация

Для переадресации к найденным в ходе выполнения запроса документам нужно установить drill атрибут для элементов.

```
<td @SetDrillRedirectionAttribute("col2", item) style="text-align:center;">@item.Value[0]</td>
```

Реализация метода SetDrillRedirectionAttribute:

```
private string SetDrillRedirectionAttribute(string coll, Row item)
{
}
```

```

        return (item.Value[0] > 0) ? string.Format("drill='{0},{1}'", coll, item.Due)
        : "";
    }
}

```

После чего на фронте приложения элемент становится кликабельным.
Для дальнейшей переадресации реализуем метод `MakeDrillAnswer`

```
public override void MakeDrillAnswer(string token) {}
```

В `string token` приходят те элементы, которые были переданы в `drill`.
Зададим поисковый критерий на поиск РК.

```
var sreq = new SRCH_REQUEST_Entity() { SRCH_KIND_NAME = "wsrch_rc" }; // сам
поисковый критерий.
```

Получим необходимые критерии для переадресации.

```

var args = token.Split(',');
var col = args[0]; // колонка
var dg = args[1]; // группа документов
var cardIsn = GetIsnsFromDues<DEPARTMENT_Entity>(Criteries.DuesCrit("regcard"));

```

Добавим критерии для переадресации.

```

sreq.SRCH_REQ_DESC_List.Add(new SRCH_REQ_DESC_Entity() { CRITERY_NAME =
"RC.DUE_DOCGROUP", VALUE = dg, DISP_VALUE = "Выбранные группы документов" });

sreq.SRCH_REQ_DESC_List.Add(new SRCH_REQ_DESC_Entity() { CRITERY_NAME =
"RC.DOC_DATE", VALUE = periodrit.VALUE, DISP_VALUE = periodrit.DISP_VALUE });

sreq.SRCH_REQ_DESC_List.Add(new SRCH_REQ_DESC_Entity() { CRITERY_NAME =
"RC.ISN_CARD_REG", VALUE = cardIsn.Replace(',', '|'), DISP_VALUE = "Выбранные
картотеки регистрации" });

```

Вызовем срабатывание переадресации.

```
Answer = new DrillParamsResult() { Result = "wsch_rc", SrchRequest = sreq };
```

Варианты реализации переадресации могут быть разными, можно заново получать необходимые ID документов из БД по SQL запросу и делать переадресацию только по этим ID или же можно записать в `drill` атрибут сразу эти ID и выполнить переадресацию.

Для SQL запроса нужно, как и в методе `FILL` получить необходимые параметры для подстановки, либо можно сделать глобальные переменные, которые уже будут хранить эти значения.

```
sreq.SRCH_REQ_DESC_List.Add(new SRCH_REQ_DESC_Entity() { CRITERY_NAME = "RC.ISN_DO
C", VALUE = string.Join("|", ids), DISP_VALUE = "равен искомому" });
```

где `ids` – список ID документов, полученных из SQL запроса или из `DRILL` атрибута.

10.1.5 Подключение отчёта к стенду

Выполняем следующие действия:

- 1) Создадим файл JSON, `_Sample.json`.
- 2) Пропишем внутри этого файла структуру:

```
{
  "Name": "Sample", - имя проекта
  "Version": "1.0.0",
  "Dependencies": {
    "Eos.Delo.DeloWebSrch": "1.0.0" - зависимости для поискового движка
  },
  "Path": "D:\\Reports\\Sample\\Sample\\bin\\Debug\\net6.0" - путь до собранного
проекта в DLL
}
```

- 3) Переходим в свою папку со стендом Дело, дальше переходим в папку Stand -> config -> d242-delosrv(название стенда) -> a -> Settings.
Пример пути: D:\\Stand\\config\\d242-delosrv\\a\\Settings
- 4) Помещаем туда созданный JSON файл.

Отчёт подключен, далее, чтобы отчёт отобразился во вкладке "Отчёты":

- 1) Выдать текущему пользователю необходимые права для доступа в отчёты.
- 2) Настройка системы.
- 3) Инструменты.
- 4) Настройка отчётов.
- 5) Нажать на кнопку регистрации отчётов.

Получаем результат добавления, на данном этапе, **отчёты записываются в БД, но не в группы!** Для того, чтобы записать отчёты в группу, нажимает кнопку «Добавить».

В результате отчёты добавляются в группу.

По умолчанию, если у отчёта в AutoReg указана группа не 1 «Все отчеты», он будет добавлен в эту группу тоже.

Даже если отчёт уже добавлен группу, он выводится в результате добавления, чтобы пользователям было понятно, какой отчёт и в какой группе зарегистрирован.

Внимание! При добавлении каждого нового отчёта, все действия нужно повторять!

11 Сборка проектов

Требования:

Dotnet SDK 6.0.11 и выше

Подготовка:

1. Скопировать локально каталог «EosSDK» (например, в "C:\"). Добавить переменную окружения EosSdk, нацеленную на локальный каталог EosSDK.

Выполнить в командной строке: `setx EosSdk EosSdkPath`

Где EosSdkPath – путь к EosSdk (например, "C:\EosSDK").

2. Скопировать локально содержимое каталога «DeloSdk\packages», NuGet пакеты системы «ДЕЛО» (например, в "C:\NuGetPackages\Eos"). Добавить источник, нацеленный на локальный каталог.

Выполнить в командной строке:

`dotnet nuget add source EosPackagesPath --name EosPackages`

Где EosPackagesPath – путь к каталогу с NuGet пакетами (например, "C:\NuGetPackages\Eos").

3. Добавить источник пакетов local_dev для публикации локально собранных NuGet пакетов.

Выполнить в командной строке:

`dotnet nuget add source LocalDevPath --name local_dev`

Где LocalDevPath – путь для результатов локальных сборок NuGet пакетов (например, "C:\NuGetPackages\local_dev").

Сборка проектов (решений):

Собрать проект (решение) с помощью команды `dotnet build`, с указанием версии сборки через свойства LicVer и PatchVer.

Выполнить в командной строке: `dotnet build SolutionPath /p:LicVer=LicVerVal /p:PatchVer=PatchVerVal`

Где SolutionPath – путь к проекту (решению), LicVerVal – версия «ДЕЛО», для которой были разработаны модули (меняется при смене версии «ДЕЛО»), PatchVerVal – последовательный номер, увеличивающийся при внесении исправлений или доработок в модули.

Например, SolutionPath = "C:\Projects\Eos.Delo.Example.Module.sln", LicVerVal = 23.7, PatchVerVal = 1

При каждой доработке значение PatchVerVal необходимо увеличивать.

Приложение А

Использование API в реальных проектах

Ниже приведены примеры использования API в реальных проектах внедрения системы «ДЕЛО».

А.1 Импорт справочников из одного сервера системы «ДЕЛО» в другой

Цель разработки:

У Заказчика есть план централизации. В частности, нужно было объединить 3 сервера системы «ДЕЛО» в один (в части справочников, пользователей. РК не трогали). Объем справочников огромный. Штатными средствами не всё можно переносить.

Краткое описание реализуемой функциональности.

Создана программа, которая может загружать данные в справочники Подразделений, Кабинетов, Групп документов, а также в модуль «Пользователи» с сохранением прав доступа, связей с кабинетами и пр.

А.2 Веб-сервис создания РКПД по данным из кадровой и других систем

Цель разработки:

В кадровой системе заводятся карточки проектов документов на отпуск и пр. Эти же данные нужны в системе «ДЕЛО», чтобы согласовать отпуск сотрудника. Разработан веб-сервис, который принимает от кадровой системы базовые параметры и создает РКПД в «ДЕЛО-WEB».

Краткое описание реализуемой функциональности:

Веб-сервис в формате JSON принимает группу документов, дату регистрации, содержание, номер связанной РК, содержимое файла и уже средствами API в «ДЕЛО-WEB» создает РКПД с заданными параметрами и создается связка с существующей РК.

А.3 Плагин "Автоматизированная подготовка и отправка уведомлений заявителю"

Цель разработки:

Отправка уведомления (электронного письма) гражданину в ответ на его обращение, зарегистрированное в системе «ДЕЛО».

Краткое описание реализуемой функциональности:

В системе «ДЕЛО» регистрируется обращение гражданина. Затем в связке регистрируется РК вида "Исходящие", которая является уведомлением заявителю. К РК уведомления приложены файлы, подписанные ЭП. Пользователь открывает РК уведомления, запускает плагин. Плагин проверяет валидность ЭП приложенных к РК файлов. Если все подписи верны, отправляет электронное письмо для гражданина, содержащее во вложении файл формата .pdf (см. Рисунок 13), сформированный по заданному шаблону, содержащий визуализацию (заранее подготовленный штамп) с информацией о том, что документ подписан ЭП.




	
АДМИНИСТРАЦИЯ ГУБЕРНАТОРА И ПРАВИТЕЛЬСТВА АЛТАЙСКОГО КРАЯ	
ДЕПАРТАМЕНТ ПО ДОКУМЕНТАЦИОННОМУ ОБЕСПЕЧЕНИЮ	
ОТДЕЛ ПО РАБОТЕ С ОБРАЩЕНИЯМИ ГРАЖДАН	
просп. Ленина, д. 59, г. Барнаул, 656035, телефон: (3852) 29-50-95, факс: (3852) 36-31-97	
13.03.2023 № Г-2/3	Героев И.Н.
	143302, Наро-Фоминск, ул. Победы, д.45, кв. 13
Уведомление	
Сообщаем, что Ваше обращение зарегистрировано 12.03.2023 за № Г-2 и направлено на рассмотрение.	
Исполнитель: Управление по основной деятельности; Филиал республики Коми.	
О результатах рассмотрения Вам сообщат письменно.	
	Генеральный директор
<div style="border: 1px solid black; padding: 5px; display: inline-block;"><div>ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ</div><hr/>Оригинал документа с ЭП хранится в системе электронного документооборота</div>	

Рисунок 13

Приложение Б

Описание интерфейсов и базовых классов ДЕЛО API 2009

Б.1 Библиотека Eos.Delo.Common

Описание элементов интерфейса IHead

IHead.BeginDbTran - начать транзакцию поставщика

```
void BeginDbTran();
```

Метод начинает транзакцию поставщика данных и открывает соединение с БД. Если транзакция уже открыта для данного поставщика, создается исключение. Завершением транзакции является её фиксация Commit() или откат Rollback(). IHead.Commit – подтвердить транзакцию

```
void Commit();
```

Метод подтверждает (фиксирует) транзакцию поставщика и закрывает соединение с БД.

IHead.Rollback – отменить транзакцию

```
void Rollback();
```

Метод отменяет (выполняет откат) транзакцию поставщика и закрывает соединение с БД.

IHead.Fill – получить результат

```
void Fill(string cmdID, TakeResult collector, params object[] args);
```

```
void Fill(string cmdID, TakeResult collector, IDictionary<string, object> namedArguments);
```

В параметре cmdID передается идентификатор команды.

В параметре collector передается накопитель результата.

В параметре args передаются аргументы.

В параметре namedArguments передаются именованные аргументы. Для именованных аргументов можно использовать объект типа CmdArguments.

IHead.Search – выполнить запрос

```
void Search(TakeResult collector, SearchArguments searchArguments);
```

В параметре collector указывается накопитель результата.

В параметре searchArguments указываются поисковый запрос, описанный с помощью объекта типа SearchArguments.

IHead.Execute – выполнить команду

```
int Execute(string cmdID, params object[] args);
```

```
int Execute(string cmdID, IDictionary<string, object> namedArguments);
```

В параметре cmdID передается идентификатор команды (хранимой процедуры).

В параметре args передаются аргументы.

В параметре namedArguments передаются именованные аргументы. Для именованных аргументов можно использовать объект типа CmdArguments.

Метод возвращает количество затронутых строк, при выполнении команды.

Описание элементов класса AppTrace

Данный класс позволяет записывать трассировочные сообщения в файл трассировки Ф3. Для того, чтобы сообщения появились в файле, должна быть выполнена настройка трассировка выполнения процессов для источника Eos.Delo.App уровня, отличного от Off.

AppTrace.Err – записать сообщение о возникновении ошибки

```
public static Exception Err(string message, params object[] args);
```

Метод выполняет запись сообщения в трассировочный файл ФЗ при настроенном уровне трассировки Error и выше.

Параметр message – строка сообщения. Строка может содержать подстановочные элементы, аналогичные используемым в методе String.Format. Параметр args – не обязательный список объектов, строковое представление значений которых нужно подставить в строку, переданную в параметре message.

Метод возвращает созданный объект типа Exception, который может быть использован для возбуждения исключения с помощью оператора throw.

AppTrace.Info – записать информационное сообщение

```
public static void Info(string message, params object[] args);
```

Метод выполняет запись сообщения в трассировочный файл процесса при настроенном уровне трассировки Information и выше. Параметры аналогичны параметрам метода Err.

AppTrace.Verbose – записать подробное информационное сообщение

```
public static void Verbose(string message, params object[] args);
```

Метод выполняет запись сообщения в трассировочный файл процесса при настроенном уровне трассировки Verbose и выше. Параметры аналогичны параметрам метода Err.

AppTrace.Warn – записать предупреждающее сообщение

```
public static void Warn(string message, params object[] args);
```

Метод выполняет запись сообщения в трассировочный файл процесса при настроенном уровне трассировки Warning и выше. Параметры аналогичны параметрам метода Err.

Описание элементов класса EntityHelper

Класс EntityHelper реализует статические методы для работы с сущностями системы.

EntityHelper.Load / EntityHelper.LoadForEdit – загрузить объект

```
public static T Load<T>(EntityBaseRef entityRef) where T : EntityBase;
```

Метод выполняет загрузку объекта по ссылке. Доступно только чтение объекта.

В параметре entityRef должен передаваться объект типа EntityBaseRef.

```
public static T LoadForEdit<T>(EntityBaseRef entityRef) where T : EntityBase;
```

Метод выполняет загрузку объекта по ссылке с возможностью редактирования объекта.

```
public static T Load<T>(EntityBaseRef entityRef, LoadOptions loadOptions) where T : EntityBase;
```

Метод выполняет загрузку объекта по ссылке с указанием опции загрузки.

В параметре loadOptions передается объект типа LoadOptions. Данный объект представляет собой опцию загрузки объекта.

```
public static IEnumerable<T> Load<T>(IEnumerable<EntityBaseRef> refs) where T : EntityBase;
```

Метод выполняет загрузку множества объектов по списку ссылок на них. Возвращает объекты в том же порядке, в каком были переданы ссылки. Выходной список

может содержать меньше элементов, если входной список содержит ссылки на не существующие объекты. Если входной список ссылок содержит дубликаты - выходной список также будет содержать дубликаты.

```
public static IEnumerable<T> Load<T>(IEnumerable<EntityBaseRef> refs, LoadOptions loadOptions) where T : EntityBase;
```

Аналогично вышеописанному методу с указанием опции загрузки.

В параметре loadOptions передается объект типа LoadOptions.

EntityHelper.Search – поиск объекта

```
public static List<ET> Search<ET>(this Query<ET> src) where ET : EntityBase;
```

```
public static List<T> Search<T>(SearchArguments args) where T : EntityBase;
```

Метод выполняет поиск объекта по запросу или заданным аргументам запроса.

Метод возвращает список объектов типа, по которому осуществлялся поиск.

EntityHelper.SearchRef – поиск ссылки объекта

```
public static List<T> SearchRef<T>(SearchArguments args) where T : EntityBaseRef;
```

Метод выполняет поиск ссылки объекта по заданным аргументам запроса.

Метод возвращает список ссылок типа, по которому осуществлялся поиск.

Описание элементов класса Facade (Eos.Delo.App)

Facade.Open – подключение поставщика данных

```
public static void Open(string aliasName);
```

```
public static void Open(string aliasName, string logId, string logPass);
```

Метод реализует подключение поставщика данных.

В параметре aliasName передается ключ настройки, может быть пустой строкой “”.

В параметрах logId передается имя пользователя, logPass – пароль пользователя, от имени которого будут производиться операции с БД.

Facade.SetUserIsn – установить пользователя для текущего контекста соединения с БД

```
public static void SetUserIsn(int userIsn);
```

Метод устанавливает пользователя для текущего контекста соединения с БД, открытого ранее с помощью Facade.Open. В параметре userIsn передается Isn пользователя системы.

Описание элементов класса LoadOptions

LoadOptions.Reload

```
public static LoadOptions Reload;
```

Поле, отвечающее за опцию загрузки объекта без учета кэша.

LoadOptions.ForEdit

```
public static LoadOptions ForEdit;
```

Поле, отвечающее за опцию загрузки объекта с возможностью редактирования объекта.

Описание элементов класса WorkContext

Рабочий контекст приложения.

```
public static WorkContext Current { get; }
```

Свойство, содержащее текущий контекст приложения.

```
public ICache Cache { get; }
```

Свойство, содержащее кэш данных.

```
public IHead Head { get; }
```

Свойство, содержащее поставщик данных.

```
public string UserID { get; }
```

Свойство, содержащее идентификатор текущего пользователя.

WorkContext.DropCurrent – уничтожить рабочий контекст

```
public static void DropCurrent();
```

Метод уничтожает ненужный рабочий контекст.

Описание элементов класса XmlSerializationHelper

XmlSerializationHelper.Deserialize – десериализовать xml

```
public static T Deserialize<T>(System.Xml.XmlReader reader);
```

Метод десериализует xml-документ. Возвращает объект указанного типа T. Параметр reader – содержащий xml-документ для десериализации.

XmlSerializationHelper.DeserializeFromFile – десериализовать xml из файла

```
public static T DeserializeFromFile<T>(string filename);
```

```
public static T DeserializeFromFile<T>(string filename,  
System.Xml.Serialization.XmlSerializer serializer);
```

Метод десериализует xml-документ из файла. Возвращает объект указанного типа T.

Параметры filename – полное имя файла для десериализации, serializer – объект класса XmlSerializer.

XmlSerializationHelper.DeserializeFromXml – десериализовать xml из xml файла

```
public static T DeserializeFromXml<T>(string xml);
```

Метод десериализует XML-документ из xml файла. Возвращает объект указанного типа T. Параметр xml – xml файл.

XmlSerializationHelper.SerializeToFile – сериализовать в файл

```
public static void SerializeToFile(object o, string filename,  
System.Xml.Serialization.XmlSerializerNamespaces xsn, Encoding encoding);
```

```
public static void SerializeToFile(object o, string filename,  
System.Xml.Serialization.XmlSerializerNamespaces xsn);
```

```
public static void SerializeToFile(object o, string filename, Encoding encoding);
```

```
public static void SerializeToFile(object o, string filename);
```

Метод сериализует указанный объект и записывает xml-документ в файл. Параметры o – объект для сериализации, filename – полное имя файла для записи xml объекта, encoding – кодировка символов, xsn – объект класса XmlSerializerNamespaces.

XmlSerializationHelper.SerializeToXml – сериализовать в файл xml

```
public static string SerializeToXml(object o,  
System.Xml.Serialization.XmlSerializerNamespaces xsn);
```

```
public static string SerializeToXml(object o, Encoding encoding);
```

```
public static string SerializeToXml(object o);
```

Метод сериализует объект в xml-строку. Параметры o – объект для сериализации, encoding – кодировка символов, xsn – объект класса XmlSerializerNamespaces.

Описание элементов класса XmlValidator

```
public List<Exception> Errors { get; }
```

Свойство, содержит список ошибок валидации.

```
public string ErrorsText { get; }
```

Свойство, позволяет получить ошибки валидации в текстовом виде: каждая ошибка с новой строки.

```
public bool Validate(string strXmlDoc);
```

Метод валидации xml документа. Возвращает результат валидации успешно или нет – true/false. Параметр strXmlDoc – xml файл.

```
public bool Validate(string strXmlDoc, string strShema);
```

Метод валидации xml документа по схеме. Возвращает результат валидации успешно или нет – true/false. Параметры strXmlDoc – xml файл, strShema – XSD схема.

```
public bool Validate(string strXmlDoc, string[] strSchemas);
```

Метод валидации xml документа по схеме, состоящей из нескольких частей. Возвращает результат валидации успешно или нет – true/false. Параметры strXmlDoc – xml файл, strSchemas - XSD схемы.

Другие элементы библиотеки

```
public enum LoadMode { Full, Short, Table, Count, Exist }
```

Перечисление типов загрузки данных:

Full – полная загрузка сущностей, с дочерними, по результатам поиска.

Short – краткая информация, без полной загрузки сущности, содержит ссылку Ref на сущность.

Table – загрузка сущности по результатам поиска, содержит значения из таблицы БД с основными свойствами сущности.

Count – количество записей, удовлетворяющих критериям поиска, количество отражается в свойстве q.SA.TotalRecords, после выполнения поиска, где q – экземпляр класса Query<...>.

Exist – существование записей, удовлетворяющих критериям поиска. Если q.SA.TotalRecords == 1, в БД есть такие записи, если q.SA.TotalRecords == 0, таких записей нет.

Б.2 Библиотека Eos.Delo.Utils

Описание элементов класса JsonConverter

JsonConverter.Deserialize – десериализовать JSON

```
public static T Deserialize<T>(string st);
```

Метод десериализует JSON в указанный тип .NET. Возвращает десериализованный объект из строки JSON.

JsonConverter.Serialize – сериализовать в JSON

```
public static string Serialize(object obj);
```

Метод сериализует указанный объект в строку JSON. Возвращает строковое представление объекта в формате JSON.

Описание элементов класса FileHelper

Все методы класса работают с файлами и директориями файловой системы.

FileHelper.CopyDirectory – копирование директории

```
public static void CopyDirectory(string source, string dest);
```

Метод копирует все файлы и папки из месторасположения source в месторасположение dest.

FileHelper.DeleteDirectory / TryDeleteDirectory – удаление директории

```
public static void DeleteDirectory(string name);
```

```
public static bool TryDeleteDirectory(string name);
```

Методы выполняют удаление директории. TryDeleteDirectory возвращает результат удаления успешно или с ошибкой - true/false. Параметр name – полное имя директории.

FileHelper.DeleteFile / TryDeleteFile – удаление файла

```
public static void DeleteFile(string name);
```

```
public static bool TryDeleteFile(string name);
```

Методы выполняют удаление файла. TryDeleteFile возвращает результат удаления успешно или с ошибкой - true/false. Параметр name – полное имя файла.

Б.3 Библиотека Eos.Delo.Wapi

Описание элементов класса MailManHelper

Класс, содержащий методы для отправки и получения email сообщений. В конструкторе класса указываются настройки для подключения к SMTP/POP3 серверу типа SmtпSettings/PopSettings.

```
public void GetEmail(string emailStorageDir, out string emailSourcePath);
```

Метод выполняют выгрузку (получение) email сообщения с сервера POP3 в папку файловой системы и удаляет с сервера. При возникновении ошибок в ходе выполнения метода, вызывается исключение. Параметры emailStorageDir – путь к папке для выгрузки email сообщений, emailSourcePath – строка, в результате выполнения метода, содержащая путь к email сообщению или null.

```
public void SendEmail(string eml, bool sendEncrypted, bool sendSigned);
```

Метод выполняют отправку email сообщения с сервера SMTP. При возникновении ошибок в ходе выполнения метода, вызывается исключение. Параметры eml – путь к email сообщению или строковое содержимое “.eml” файла, sendEncrypted – необходимость отправки в зашифрованном виде, sendSigned – необходимость отправки с цифровой подписью.

Б.4 Библиотека Eos.Delo96.Api

Описание элементов интерфейса IFileHelper (Eos.Delo96.Api.Entities.Helpers)

Интерфейс наследуется классами-помощниками РК документа (DOC_RC_Helper), резолюции (RESOLUTION_Helper), РКПД (PRJ_RC_Helper). Все методы выполняются для экземпляра одного из таких классов.

```
REF_FILE_Entity Append_REF_FILE(string fileNameWithExtension, string category);
```

Метод добавляет файл к сущности. Возвращает сущность REF_FILE_Entity добавленного файла.

Параметры: `fileNameWithExtension` – имя файла, `category` – наименование категории.

```
REF_FILE_ACCESS_Entity Append_REF_FILE_ACCESS(REF_FILE_Ref fileRef, DEPARTMENT_Ref clRef);
```

Метод добавляет доступ к файлу для должностного лица. Возвращает созданную сущность `REF_FILE_ACCESS_Entity` доступа к файлу. Параметры: `fileRef` – ссылка на файл, `clRef` – ссылка на ДЛ.

```
REF_FILE_EDS_Entity Append_REF_FILE_EDS(REF_FILE_Ref fileRef);
```

Метод добавляет ЭП для файла. Возвращает сущность `REF_FILE_EDS_Entity` ЭП файла.

Параметр `fileRef` – ссылка на файл.

```
void ApplyDefaultRefFileRules(REF_FILE_Entity fileRef);
```

Метод добавляет правила по умолчанию для файла.

Параметр `fileRef` – сущность файла.

```
bool CanAddFile(string category);
```

Проверка прав на добавление файла. `True` – есть права, `false` – нет прав.

```
bool CanCheckSignFile(REF_FILE_Ref fileRef);
```

Проверка прав для проверки подписи файла. `True` – есть права, `false` – нет прав.

```
bool CanDeleteFile(REF_FILE_Ref fileRef);
```

Проверка прав для удаления файла. `True` – есть права, `false` – нет прав.

```
bool CanEditFileContent(REF_FILE_Ref fileRef);
```

Проверка прав для редактирования файла. `True` – есть права, `false` – нет прав.

```
bool CanEditFileDescription(REF_FILE_Ref fileRef);
```

Проверка прав для редактирования описания файла. `True` – есть права, `false` – нет прав.

```
bool CanEditFileWeight(string category);
```

Проверка прав для изменения порядка следования файлов. `True` – есть права, `false` – нет прав.

```
bool CanSignFile(REF_FILE_Ref fileRef);
```

Проверка прав для подписи файла. `True` – есть права, `false` – нет прав.

```
bool CanViewFile(REF_FILE_Ref fileRef);
```

Проверка прав для просмотра файла. `True` – есть права, `false` – нет прав.

```
IEnumerable<REF_FILE_Entity> GetFiles(bool visibleOnly, params string[] categories);
```

Метод для получения файлов. Метод возвращает коллекцию сущностей файлов, относящихся к экземпляру объекта, для которого метод был вызван.

Описание элементов класса `ClassifHelper` (`Eos.Delo96.Api.Entities.Helpers`)

Класс содержит только статические методы.

`ClassifHelper.Contact2Department` – получить ссылку на подразделение (ДЛ)

```
public static DEPARTMENT_Ref Contact2Department(CONTACT_Ref contactRef);
```


Метод возвращает ссылку на подразделение (ДЛ) по ссылке на контакт в организации. Параметр метода: `contactRef` – ссылка на контакт в организации.

`ClassifHelper.DueDep2DueOrganiz` – получить due организации

```
public static string DueDep2DueOrganiz(string dueDep);
```

Метод возвращает due организации по due подразделения (ДЛ), связанным с ней. Параметр метода: `dueDep` – due подразделения (ДЛ).

`ClassifHelper.GetARDescriptList` – получить доп. реквизитов группы документов

```
public static List<AR_DESCRIPT_Entity> GetARDescriptList(DOCGROUP_CL_Ref docGroupRef);
```

```
public static List<AR_DESCRIPT_Entity> GetARDescriptList(DOCGROUP_CL_Entity docGroup);
```

Метод возвращает список доп. реквизитов для группы документов. Параметр метода: `docGroupRef` – ссылка на группу документов, или `docGroup` – сущность группы документов.

`ClassifHelper.GetDepartmentLinkFromDepartment` – получить сущность подразделения (ДЛ)

```
public static DEPARTMENT_Entity GetDepartmentLinkFromDepartment(string departmentDue);
```

Метод возвращает сущность подразделения (ДЛ) по указанному due. Параметр метода: `departmentDue` – due подразделения (ДЛ).

`ClassifHelper.GetDepartmentLinkFromOrganization` – получить сущность подразделения (ДЛ)

```
public static DEPARTMENT_Entity GetDepartmentLinkFromOrganization(string orgDue);
```

Метод возвращает сущность подразделения (ДЛ) по указанному due организации, связанной с ним. Параметр метода: `orgDue` – due организации.

`ClassifHelper.GetLinkedOrganizForDep` – получить сущность организации

```
public static ORGANIZ_CL_Entity GetLinkedOrganizForDep(string dep_due);
```

Метод возвращает сущность организации по указанному due подразделения (ДЛ), связанным с ней. Параметр метода: `dep_due` – due подразделения (ДЛ).

`ClassifHelper.GetOrganizationLinkFromDepartment` – получить ссылку на организацию

```
public static ORGANIZ_CL_Ref GetOrganizationLinkFromDepartment(string departmentDue);
```

Метод возвращает ссылку на организацию по указанному due подразделения (ДЛ), связанным с ней. Параметр метода: `departmentDue` – due подразделения (ДЛ).

`ClassifHelper.GetRubricARDescriptList` – получить доп. реквизиты рубрик группы документов

```
public static List<AR_DESCRIPT_Entity> GetRubricARDescriptList(DOCGROUP_CL_Ref docGroupRef);
```

```
public static List<AR_DESCRIPT_Entity> GetRubricARDescriptList(DOCGROUP_CL_Entity docGroup);
```

Метод возвращает список доп. реквизитов рубрик для указанной группы документов. Параметр метода: `docGroupRef` – ссылка на группу документов, или `docGroup` – сущность группы документов.

`ClassifHelper.RefByIsn` – получить ссылку на сущность

```
public static T RefByIsn<T>(int isn) where T : EntityBaseRef;
```

Метод возвращает ссылку на сущность типа T. Параметр метода: isn – isn сущности.

Описание элементов класса ClassifEntity (Eos.Delo96.Api.Entities)

Абстрактный класс, который наследуют сущности справочников. Данный класс является наследником класса EntityBase.

```
public override EntityBaseRef OwnerRef { get; }
```

Свойство, содержащее ссылку на родительскую сущность.

```
public virtual int ISN_LCLASSIF { get; set; }
```

Свойство, содержащее идентификатор ISN элемента.

```
public virtual int? WEIGHT { get; set; }
```

Свойство, содержащее вес элемента.

```
public virtual string CLASSIF_NAME { get; set; }
```

Свойство, содержащее наименование элемента.

```
public virtual int PROTECTED { get; set; }
```

Свойство, содержащее признак запрета удаления элемента.

```
public virtual int DELETED { get; set; }
```

Свойство, содержащее признак логического удаления элемента.

```
public virtual string NOTE { get; set; }
```

Свойство, содержащее примечание элемента.

Описание элементов класса ClassifTree (Eos.Delo96.Api.Entities)

Абстрактный класс, который наследуют сущности справочников. Данный класс является наследником класса ClassifEntity.

```
public override int ISN_LCLASSIF { get; set; }
```

Свойство, содержащее идентификатор ISN элемента.

```
public virtual string PARENT_DUE { get; set; }
```

Свойство, содержащее DUE родительской записи элемента.

```
public abstract EntityBaseRef ParentRef { get; }
```

Свойство, содержащее ссылку на родительскую сущность.

```
public virtual string DUE { get; set; }
```

Свойство, содержащее DUE элемента.

```
public virtual int ISN_NODE { get; set; }
```

Свойство, содержащее ISN элемента.

```
public virtual int IS_NODE { get; set; }
```

Свойство, содержащее признак вершины (0 – вершина; 1 – лист).

```
public virtual int? ISN_HIGH_NODE { get; set; }
```

Свойство, содержащее ISN вышестоящей вершины.

Описание элементов класса CUSTOM_STORAGE_Helper (Eos.Delo96.Api.Entities.Helpers)

CUSTOM_STORAGE_Helper.Append – добавить записи, без удаления существующих

```
public static void Append(string valueId, IEnumerable<string> values, int ownerKind, string ownerId);
```

Метод добавляет записи в таблицу. Если для данного valueId уже имеются записи в таблице, то новые записи будут добавлены в конец списка уже имеющихся записей, т.е. с порядковым номером (ordernum) выше, чем последней записи с таким valueId.

CUSTOM_STORAGE_Helper.Delete – удалить запись

```
public static void Delete(string valueId);
```

Метод удаляет все записи в таблице с заданным параметром valueId.

CUSTOM_STORAGE_Helper.FindByID / FindByOwner – найти запись

```
public static List<CUSTOM_STORAGE_ID_Entity> FindByID(string valueId);
```

Метод выполняет поиск по идентификатору значения (valueId) с использованием оператора LIKE. Возвращает список сущностей CUSTOM_STORAGE_ID_Entity.

```
public static List<CUSTOM_STORAGE_ID_Entity> FindByID(string valueId, string value);
```

Метод выполняет поиск по идентификатору значения (valueId) и значению (value) с использованием оператора LIKE. Возвращает список сущностей CUSTOM_STORAGE_ID_Entity.

```
public static List<CUSTOM_STORAGE_ID_Entity> FindByOwner(int ownerKind, string ownerId);
```

Метод выполняет поиск по объекту - владельцу значения. Возвращает список сущностей CUSTOM_STORAGE_ID_Entity.

CUSTOM_STORAGE_Helper.Load / LoadChunkBatch / LoadSingle – загрузить записи

```
public static List<string> Load(string valueId);
```

Метод выполняет загрузку списка значений для указанного valueId.

```
public static string LoadChunkBatch(string valueId);
```

Метод выполняет загрузку значения для указанного valueId, которое хранится в нескольких записях таблицы БД.

Используется для единственного значения valueId, длина которого может превышать 2000 символов.

```
public static string LoadSingle(string valueId);
```

Метод выполняет загрузку единственного значения для указанного valueId.

CUSTOM_STORAGE_Helper.Save / SaveChunkBatch / SaveSingle – сохранить записи

```
public static List<string> Save(string valueId, IEnumerable<string> values, int ownerKind, string ownerId);
```

Метод выполняет сохранение коллекции значений. Старое значение по valueId удаляется.

Параметры: valueId – идентификатор записи, values – коллекция значений, ownerKind - тип объекта владельца, ownerId - идентификатор объекта владельца.

Метод возвращает старое значение по идентификатору записи, если оно имелось.

```
public static string SaveChunkBatch(string valueId, string value, int ownerKind,
string ownerId);
```

Метод выполняет сохранение значения, длина которого превышает 2000 символов. Старое значение по valueId удаляется.

Параметры: valueId – идентификатор записи, value – значение, ownerKind - тип объекта владельца, ownerId - идентификатор объекта владельца.

Метод возвращает старое значение по идентификатору записи, если оно имелось.

```
public static string SaveSingle(string valueId, string value, int ownerKind, string
ownerId);
```

Метод выполняет сохранение единственного значения. Старое значение по valueId удаляется.

Параметры: valueId – идентификатор записи, value – значение, ownerKind - тип объекта владельца, ownerId - идентификатор объекта владельца.

Метод возвращает старое значение по идентификатору записи, если оно имелось.

Описание элементов класса DOC_RC_Helper (Eos.Delo96.Api.Entities.Helpers)

```
public ACQUAINTANCE_Entity Append_ACQUAINTANCE(DEPARTMENT_Ref clRef);
```

Метод добавляет запись в журнале ознакомления. Параметр clRef – ссылка на ДЛ.

```
public DOC_EXE_Entity Append_DOC_EXE(DEPARTMENT_Ref depRef);
```

Метод добавляет ДЛ «исполнители». Параметр depRef – ссылка на ДЛ.

```
public DOC_SIGN_Entity Append_DOC_SIGN(DEPARTMENT_Ref depRef);
```

Метод добавляет ДЛ «подписал». Параметр depRef – ссылка на ДЛ.

```
public DOC_WHO_Entity Append_DOC_WHO(DEPARTMENT_Ref depRef);
```

Метод добавляет ДЛ «кому». Параметр depRef – ссылка на ДЛ.

```
public FORWARD_Entity Append_FORWARD(DEPARTMENT_Ref depRef);
```

Метод добавляет запись в журнал пересылки документа. Параметр depRef – ссылка на ДЛ.

```
public JOURNAL_Entity Append_JOURNAL(REF_SEND_Entity snd);
```

Метод добавляет запись в журнал передачи документов. Параметр snd – ссылка на адресата.

```
public JOURNAL_Entity Append_JOURNAL(int recType, EntityBaseRef clRef,
DuplicateEnumerator duplicator);
```

Метод добавляет запись в журнал передачи документов. Параметр recType – тип записи (0 – кандидат, 1 – передача, 2 – списание в дело, 3 – уничтожение), clRef – в зависимости от типа записи, объект типа NOMENKL_CL_Ref или DEPARTMENT_Ref, duplicator – объект типа DuplicateEnumerator.

```
public REF_ACCESS_CARD_Entity Append_REF_ACCESS_CARD(DEPARTMENT_Ref cardRef);
```

Метод добавляет принадлежность РК к картотеке, cardRef – ссылка типа DEPARTMENT_Ref.

```
public REF_CORRESP_Entity Append_REF_CORRESP(CONTACT_Ref clRef, int correspKind);
```

Метод добавляет корреспондента или сопроводительный документ к РК. Параметры `clRef` – ссылка типа `CONTACT_Ref`, `correspKind` – вид записи (1 – корреспонденты, 2 – сопроводительные документы).

```
public REF_FILE_Entity Append_REF_FILE(string fileNameWithExtension, string category);
```

Метод добавляет файл к РК. Параметры `fileNameWithExtension` – имя файла с расширением, `category` – категория файла.

```
public REF_FILE_ACCESS_Entity Append_REF_FILE_ACCESS(REF_FILE_Ref fileRef,
DEPARTMENT_Ref clRef);
```

Метод добавляет доступ к файлу РК для ДЛ. Параметры `fileRef` – ссылка на файл, `clRef` – ссылка на подразделение (ДЛ).

```
public REF_FILE_EDS_Entity Append_REF_FILE_EDS(REF_FILE_Ref fileRef);
```

Метод добавляет ЭП файла РК. Параметр `fileRef` – ссылка на файл.

```
public REF_LETTER_Entity Append_REF_LETTER(CITIZEN_Ref clRef);
```

Метод добавляет заявителя письма к РК. Параметр `clRef` – ссылка на гражданина.

```
public REF_LINK_Entity Append_REF_LINK(LINK_CL_Ref linkRef, EntityBaseRef linkedRef);
```

Метод добавляет связку к РК. Параметр `linkRef` – ссылка на связку, `linkedRef` – ссылка на связываемую РК или РКПД.

```
public REF_RUBRIC_Entity Append_REF_RUBRIC(RUBRIC_CL_Ref clRef);
```

Метод добавляет рубрику к РК. Параметр `clRef` – ссылка на рубрику.

```
public REF_SEND_Entity Append_REF_SEND(CITIZEN_Ref clRef);
```

Метод добавляет адресата к РК. Параметр `clRef` – ссылка на гражданина.

```
public REF_SEND_Entity Append_REF_SEND(PRJ_REF_SEND_Entity rs);
```

Метод добавляет адресата к РК. Параметр `rs` – адресат из РКПД.

```
public REF_SEND_Entity Append_REF_SEND(CONTACT_Ref clRef, bool FillContact);
```

```
public REF_SEND_Entity Append_REF_SEND(CONTACT_Ref clRef);
```

Метод добавляет адресата к РК. Параметр `clRef` – ссылка на контакт организации.

```
public REF_SOISP_Entity Append_REF_SOISP(CONTACT_Ref clRef);
```

Метод добавляет соисполнителя документа. Параметр `clRef` – ссылка на контакт организации.

```
public REF_VISA_Entity Append_REF_VISA(DEPARTMENT_Ref depRef);
```

Метод добавляет визу РК. Параметр `depRef` – ссылка на подразделение (ДЛ).

```
public SEV_REPORT_Entity Append_SEV_REPORT(CITIZEN_Ref citRef);
```

Метод добавляет доклад. Параметр `citRef` – ссылка на гражданина.

```
public SEV_REPORT_Entity Append_SEV_REPORT(ORGANIZ_CL_Ref orgRef);
```

Метод добавляет доклад. Параметр `orgRef` – ссылка на организацию.

```
public void ApplyDefaultRules();
```

Метод применяет к документу РК правила заполнения полей и разделов по умолчанию.

```
public void ApplyDeliveryRule(int prev_delivery);
```

Метод заполняет вид доставки входящего документа с учетом значений по умолчанию и параметра prev_delivery – isn вида доставки.

```
public DOC_RC_Entity Create(DOCGROUP_CL_Ref dgRef);
```

Метод создает РК определенной группы документов. Параметр dgRef – due группы документов.

```
public RESOLUTION_Helper CreateResolutionHelper();
```

Метод создаёт экземпляр класса-помощник резолюции.

```
public RESOLUTION_Helper CreateResolutionHelper(RESOLUTION_Entity resolution);
```

Метод создаёт экземпляр класса-помощник резолюции с указанием резолюции. Параметр resolution – сущность резолюции.

```
public REF_SEND_Entity Fill_REF_SEND(REF_SEND_Entity result, bool FillContact);
```

```
public IEnumerable<REF_FILE_Entity> GetFiles(bool visibleOnly, params string[] categories);
```

Метод для получения файлов РК. Параметр visibleOnly – только видимые файлы, categories – категории.

```
public List<RESOLUTION_Entity> GetResolutions();
```

Метод для получения резолюций РК. Возвращает список резолюций.

```
public List<RESOLUTION_Entity> GetResolutions(string cardDue);
```

Метод для получения резолюций РК, принадлежащих определенной картотеке. Параметр cardDue – due картотеки. Метод возвращает список резолюций.

```
public bool ValidateMandatoryRules(DocMandatoryRules validateRules);
```

```
public bool ValidateMandatoryRules(DocMandatoryRules validateRules, List<string> errors);
```

Метод проверяет заполненность полей и разделов в соответствии с правилами. Параметры validateRules – перечень правил для проверки, errors – список, для накопления ошибок. Метод возвращает false – если нарушено одно из правил, проверка которых заказана в параметре validateRules.

```
public bool ValidateSpecimen(List<string> errors);
```

Метод проверяет экземплядность РК и их использование. Параметр errors – накопитель ошибок.

```
public enum DocMandatoryRules { ALL, RC, SEND, CORRESP, SIGN, WHO, FILE, AR, EXE, RUBRIC }
```

Перечисление правил проверки.

Описание элементов класса PRJ_RC_Helper (Eos.Delo96.Api.Entities.Helpers)

```
public PRJ_EXEC_Entity Append_PRJ_EXEC(DEPARTMENT_Ref clRef);
```

Метод добавляет исполнителя РКПД. Параметр clRef – ссылка на подразделение (ДЛ).

```
public PRJ_FORUM_Entity Append_PRJ_FORUM();
```

Метод добавляет обсуждение. Возвращает сущность PRJ_FORUM_Entity.

```
public PRJ_REF_RUBRIC_Entity Append_PRJ_REF_RUBRIC(RUBRIC_CL_Ref clRef);
```

Метод добавляет рубрику. Параметр clRef – ссылка на рубрику.

```
public PRJ_REF_SEND_Entity Append_PRJ_REF_SEND(CONTACT_Ref clRef);
```

Метод добавляет адресата РКПД. Параметр clRef – ссылка на контакт организации.

```
public PRJ_REF_SEND_Entity Append_PRJ_REF_SEND(CITIZEN_Ref clRef);
```

Метод добавляет адресата РКПД. Параметр clRef – ссылка на гражданина.

```
public PRJ_REF_SEND_Entity Append_PRJ_REF_SEND(DEPARTMENT_Ref clRef);
```

Метод добавляет адресата РКПД. Параметр clRef – ссылка на подразделение (ДЛ).

```
public PRJ_VISA_SIGN_Entity Append_PRJ_VISA_SIGN(DEPARTMENT_Ref clRef, DEPARTMENT_Ref parentRef, int kind);
```

Метод добавляет подпись или визу РКПД. Параметр clRef – ссылка на ДЛ, parentRef – ссылка на ДЛ (может быть null), kind – вид записи (1 – подпись, 2 – виза).

```
public REF_FILE_Entity Append_REF_FILE(string fileNameWithExtension, string category);
```

Метод добавляет файл к РКПД. Параметры fileNameWithExtension – имя файла с расширением, category – категория файла.

```
public REF_FILE_ACCESS_Entity Append_REF_FILE_ACCESS(REF_FILE_Ref fileRef, DEPARTMENT_Ref clRef);
```

Метод добавляет доступ к файлу для должностного лица. Возвращает созданную сущность REF_FILE_ACCESS_Entity доступа к файлу. Параметры: fileRef – ссылка на файл, clRef – ссылка на ДЛ.

```
public REF_FILE_EDS_Entity Append_REF_FILE_EDS(REF_FILE_Ref fileRef);
```

Метод добавляет ЭП для файла. Возвращает сущность REF_FILE_EDS_Entity ЭП файла. Параметр fileRef – ссылка на файл.

```
public REF_LINK_Entity Append_REF_LINK(LINK_CL_Ref linkRef, EntityBaseRef linkedRef);
```

Метод добавляет связку к РКПД. Параметр linkRef – ссылка на связку, linkedRef – ссылка на связываемую РК или РКПД.

```
public SEV_REPORT_Entity Append_SEV_REPORT(ORGANIZ_CL_Ref orgRef);
```

Метод добавляет доклад. Параметр orgRef – ссылка на организацию.

```
public void ApplyDefaultRefFileRules(REF_FILE_Entity rf);
```

Метод добавляет правила по умолчанию для файла. Параметр rf – сущность файла.

```
public void ApplyDefaultRules();
```

Метод добавляет правила по умолчанию для РКПД.

```
public void ApplyMainExecutor();
```

Метод устанавливает главного исполнителя РКПД.

```
public PRJ_RC_Entity Create(DOCGROUP_CL_Ref dgRef);
```

Метод создает РКПД определенной группы документов. Параметр dgRef – ссылка на группу документов.

```
public PRJ_RC_Entity GetCurrentVersion();
```

Метод для получения на текущей версии РКПД.

```
public PRJ_RC_Ref GetCurrentVersionRef();
```

Метод для получения ссылки на текущую версию РКПД.

```
public IEnumerable<REF_FILE_Entity> GetFiles(bool visibleOnly, params string[] categories);
```

Метод для получения файлов РКПД. Параметр visibleOnly – только видимые файлы, categories – категории.

```
public bool ValidateMandatoryRules(IEnumerable<string> validateRules, List<string> errors);
```

```
public bool ValidateMandatoryRules(PrjMandatoryRules validateRules, List<string> errors);
```

Метод проверяет заполненность полей и разделов в соответствии с правилами. Параметры validateRules – перечень правил для проверки, errors – список, для накопления ошибок. Метод возвращает false – если нарушено одно из правил, проверка которых заказана в параметре validateRules.

```
public enum PrjMandatoryRules { ALL, PRJ, SEND, VISA, SIGN, FILE, AR, RUBRIC }
```

Перечисление правил для РКПД для проверки.

Описание элементов класса RESOLUTION_Helper (Eos.Delo96.Api.Entities.Helpers)

```
public REF_FILE_Entity Append_REF_FILE(string fileNameWithExtension, string category);
```

Метод добавляет файл к резолюции. Параметры fileNameWithExtension – имя файла с расширением, category – категория файла.

```
public REF_FILE_ACCESS_Entity Append_REF_FILE_ACCESS(REF_FILE_Ref fileRef, DEPARTMENT_Ref clRef);
```

Метод добавляет доступ к файлу для должностного лица. Возвращает созданную сущность REF_FILE_ACCESS_Entity доступа к файлу. Параметры: fileRef – ссылка на файл, clRef – ссылка на ДЛ.

```
public REF_FILE_EDS_Entity Append_REF_FILE_EDS(REF_FILE_Ref fileRef);
```

Метод добавляет ЭП для файла. Возвращает сущность REF_FILE_EDS_Entity ЭП файла. Параметр fileRef – ссылка на файл.

```
public REPEAT_RES_Entity Append_REPEAT_RES();
```

Метод добавляет повторение резолюции, если его не было добавлено ранее.

```
public REPLY_Entity Append_REPLY();
```

```
public REPLY_Entity Append_REPLY_Contact(CONTACT_Ref clRef);
```

```
public REPLY_Entity Append_REPLY_Department(DEPARTMENT_Ref clRef);
```

Методы добавляют исполнителей резолюции. Параметр clRef – ссылка на подразделение (ДЛ) или контакт организации.

```
public void ApplyDefaultRefFileRules(REF_FILE_Entity rf);
```

Метод добавляет правила по умолчанию для файла резолюции. Параметр rf – файл, объект типа REF_FILE_Entity.

```
public void ApplyDefaultRules();
```

Метод добавляет правила по умолчанию для резолюции.


```
public RESOLUTION_Entity Create(int kind_resolution, RESOLUTION_Ref parentRef);
```

Метод создает резолюцию. Параметры `kind_resolution` – вид резолюции (1 – резолюция, 2 – пункт), `parentRef` – ссылка на родительскую резолюцию (может быть null).

```
public IEnumerable<REF_FILE_Entity> GetFiles(bool visibleOnly, params string[] categories);
```

Метод для получения файлов резолюции. Параметр `visibleOnly` – только видимые файлы, `categories` – категории.

Описание элементов класса UserContext

Пользовательский контекст.

```
public static UserContext Current { get; }
```

Свойство, содержащее текущий пользовательский контекст.

```
public USER_CL_Entity USER_CL { get; }
```

Свойство, содержащее текущий пользовательский контекст.

```
public void ReLoad();
```

Метод перезагружает пользовательский контекст.

Приложение В

Системные настройки

В.1 Настройки системы

Под настройками Системы понимаются ее **параметры времени выполнения**. При необходимости настройки могут быть изменены во время работы приложения без его перезапуска.

Система также использует другие виды параметров: **конфигурационные, параметры развертывания**. Данные параметры не являются настройками.

Платформа предоставляет интерфейс для хранения системных настроек.

В.2 Тип объекта

Таблица В. 1

Свойство	Значение
Название (ru)	Системные настройки
Название (en)	Application Settings
Имя API	Application Settings

В.3 Реквизиты

Таблица В. 2

APP_SETTINGS	Реквизит	Имя API	Тип	Ред?	Ноль?	Уник?	Описание
NAMESPACE	Пространство имен	Namespace	String(64)	Нет	Нет	Нет	
INSTANCE	Экземпляр	Instance	String(64)	Нет	Нет	Нет	
TYPENAME	Тип	Typename	String(64)	Нет	Нет	Нет	
PROPERTY	Свойство	Property	String(255)	Нет	Нет	Нет	
VALUE	Значение	Value	String (64000)	Нет	Нет	Нет	
DATA_TYPE	Тип данных	DataType	Enum В.4 Типы данных системных настроек	Нет	Нет	Нет	Не допускается использовать следующие простые типы: short, int, decimal, float, char, byte
TIMESTAMP	Дата обновления	Timestamp	Datetime (Local)	Нет	Нет	Нет	

В.4 Типы данных системных настроек

Использование:

Таблица В. 3

Имя API	Значение	Название	Описание	Комментарий
String	1	Строка		
Long	2	Целое число		
Bool	3	Логическое значение		
DateTime	4	Дата и время		
TimeSpan	5	Интервал времени		
Double	6	Число двойной точности		
Guid	7	Идентификатор		
Enum	8	Перечисление		Enum - это не тип, а вообще любой Enum

В.5 Функции и права системных настроек

Права для функций системных настроек определяются каждой Системой самостоятельно.

Таблица В. 4

Функция	Комментарий
Запись системных настроек	Позволяет: создать настройки, если такой настройки нет перезаписать настройки, если она уже есть в системе.
Удаление системных настроек	Для удаления настроек необходимо передать в сервис пустой объект. В контроллере передается пустой json.
Чтение системных настроек	Получает настройки по ключу.

В.6 Примеры

Для работы через **AppSettings** класс должен быть **immutable**. Допускается только использование свойств **{get; private set;}** или **{ get; init; }**.

В качестве коллекций допускается только использование **ReadOnlyList** и **ReadOnlyDictionary**.

Запись класса через интерфейс:

```
// запись класса через интерфейс
using Eos.Platform.AppSettings;
```

```
var appSettingsService =
    serviceProvider.GetRequiredService<IAppSettingsService>();
```

```
await
appSettingsService.SetAsync(someClass, new MutationContext(...), "instanceName");
```

```
//Либо
```

```
await appSettingsService.SetAsync(typeof(SomeClass),
someObject, new MutationContext(...), "instanceName");
```

Чтение класса через интерфейс:

```
// чтение класса через интерфейс
using Eos.Platform.AppSettings;
```

```
var appSettingsService =
serviceProvider.GetRequiredService<IAppSettingsService>();
var someClass = await appSettingsService.GetAsync<SomeClass>("instanceName");
```

```
//Либо
```

```
var someClass = await
appSettingsService.GetAsync(typeof(SomeClass), "instanceName");
```

Чтение словаря объектов определенного типа через интерфейс (ключ словаря - имя экземпляра):

```
// чтение словаря объектов определенного типа через интерфейс
using Eos.Platform.AppSettings;
```

```
var appSettingsService =
serviceProvider.GetRequiredService<IAppSettingsService>();
var someClassList = await appSettingsService.GetListAsync<SomeClass>();
```

```
//Либо
```

```
var someClassList = await appSettingsService.GetListAsync(typeof(SomeClass));
```

Получение копии объекта с использованием Default instance:

```
// получение копии объекта с использованием Default instance
using Eos.Platform.AppSettings;
```

```
var appSettingsService =
serviceProvider.GetRequiredService<IAppSettingsService>();
var someClass = await appSettingsService.GetAsync<SomeClass>("instance");
var defaultSomeClass = await appSettingsService.GetAsync<SomeClass>();
```

```
var copy = AppSettingsHelper.DeepCopy(someClass, defaultSomeClass);
```

В.7 Запись в AppSettings конкретного класса

Для того, чтобы открыть запись через middleware в AppSettings необходимо добавить свой **IMutationLogicMapPart** и при необходимости добавить свой **MutationLogic**.

В приведенном ниже примере мы просто открываем запись для определенного класса:

```
// запись в AppSettings конкретного класса
using Eos.Delo.Platform.Storage.Services.Logic;
using Eos.Delo.Platform.Storage.Services.Logic.MutationLogic;
using Eos.Delo.Samples.AppSettings.Module.Services.Logic.MutationLogic;
using Eos.Platform.AppSettings;
using System.Collections.Generic;

namespace Eos.Delo.Samples.AppSettings.Module.Services.Logic
{
    public class MutationLogicMapPartAppSettingsTest : IMutationLogicMapPart
    {
        private static readonly List<MutationLogicMapPartItem> _supportedTypes
= new()
        {
            new((obj) =>
            {
                var appSettings = (IAppSettings)obj;

                if (appSettings.Namespace
== typeof(AppSettingsTestClass).Namespace && appSettings.TypeName ==
nameof(AppSettingsTestClass))
                    return typeof(PermissiveMutationLogic);
                else
                    return typeof(DeniedMutationLogic);
            }, new[] { typeof(IAppSettings) })
        };

        public IReadOnlyList<MutationLogicMapPartItem> SupportedTypes =>
            _supportedTypes;
    }
}

//Добавление в startup:

serviceCollection.AddSingleton<IMutationLogicMapPart,
MutationLogicMapPartAppSettingsTest>();
```

Также можно определить собственный MutationLogic, если нужно описать права.

Примеры можно найти в дистрибутиве системы в каталоге «Sdk» (проект Eos.Delo.Samples.AppSettings.Module).

В.8 Работа с Secret

Позволяет сохранять пароли в AppSettings.

Допустим у нас есть класс с полем Secret:

```
public class TestPass
{
    public Secret Password { get; init; }
```

```
}
```

Тогда работа с паролями может выглядеть так:

```
// работа с паролями
```

```
public async Task SampleMethod(IServiceProvider serviceProvider)
{
    using var scope = serviceProvider.CreateScope();
    var provider = scope.ServiceProvider;
    var db = provider.GetRequiredService<IDataContext>();
    var appSettingsService =
provider.GetRequiredService<IAppSettingsService>();
    var secretStore = provider.GetRequiredService<ISecretStore>();

    var userService =
provider.GetRequiredService<UserCachingService>();
    var user = await userService.GetUserAsync(3611);

    var context = MiddlewareHelper.CreateMutationContext(db, user);

    var testPass = new TestPass()
    {
        Password = new Secret()
        {
            Value = "11111"
        }
    };
    await appSettingsService.SetAsync(testPass, context);

    var secretHandler = await appSettingsService.GetAsync<testPass>();
    var pass = await
secretStore.GetValueAsync(secretHandler.Password);
}
```

Пример SecretSampleClass.cs:

```
// SecretSampleClass.cs
```

```
using Eos.Delo.Platform.Storage.Helpers;
using Eos.Delo.Platform.Storage.Model;
using Eos.Delo.Platform.Storage.Services.Caching;
using Eos.Platform.AppSettings;
using Microsoft.Extensions.DependencyInjection;
using System;
using System.Threading.Tasks;

namespace Eos.Delo.Samples.AppSettings.Module
{
    public class SecretSampleClass
    {
        public async Task SampleMethod(IServiceProvider serviceProvider)
        {
            using var scope = serviceProvider.CreateScope();
            var provider = scope.ServiceProvider;
            var db = provider.GetRequiredService<IDataContext>();
            var appSettingsService = provider.GetRequiredService<IAppSettingsS
```

```

ervice>();
    var secretStore = provider.GetRequiredService<ISecretStore>();

    var userService = provider.GetRequiredService<UserCachingService>(
);
    var user = await userService.GetUserAsync(3611);

    var context = MiddlewareHelper.CreateMutationContext(db, user);

    var key = await secretStore.SetValueAsync("11111");
    await appSettingsService.SetAsync(new AppSettingsSecret() { Key =
key }, context);

    var secretHandler = await appSettingsService.GetAsync<AppSettingsS
ecret>();
    var pass = await secretStore.GetValueAsync(secretHandler);
    }
}
}

```

В.9 Чтение и запись через API

Для возможности работы с классом через API необходимо зарегистрировать **AppSettingsTypeMappingServicePart** с указанием типов:

```

serviceCollection.AddSingleton(new AppSettingsTypeMappingServicePart(new
HashSet<Type>() { typeof(Carma.Carma) }));

```

Прототип контроллера есть в Eos.Platform.AppSettings, но конечный адрес и сам контроллер физически определяются в конечной системе. Соответственно тут примем адрес контроллера = {name}

Таблица В. 5

Описание	Параметры	Метод	Возвращает	Описание
{name}?namespace={0}&typename={1}&instance={2}&mergedefault={3}	namespace, typename, instance (по умолчанию - "Default") (все string), mergedefault (по умолчанию - true, bool) (Пространство имен, тип, экземпляр)	Get	File (json)	Метод для получения определенного класса в виде json. Если параметр mergedefault - true, метод get возвращает полную копию объекта instance, но если в этом объекте какие-то поля не заполнены, они будут по умолчанию взяты из объекта instance="Default"
{name}/upload?namespace={0}&typename={1}&instance={2}	namespace, typename, instance (по умолчанию - "Default") (все string) (Пространство имен, тип, экземпляр) Передача класса в виде json как StreamContent в	Post		Метод для загрузки класса в систему.

Описание	Параметры	Метод	Возвращает	Описание
	Body			
{name}/get-list?namespace={0}&typename={1}&mergedefault={2}	namespace, typename, (все string), mergedefault (по умолчанию - true, bool) (Пространство имен, тип)	Get	File (json)	Метод для получения словаря объектов определенного типа в виде json (ключ словаря - имя экземпляра). Если параметр mergedefault - true, метод get-list возвращает полную копию объектов словаря, но если в любом объекте списка какие-то поля не заполнены, они будут по умолчанию взяты из объекта instance="Default"

Приложение Г

Объекты системы

ACQUAINTANCE

Список ознакомлений:

Таблица Г. 1

Имя	Тип	Комментарии
ISN_DOC	Целое число	ISN документа
KIND_DOC	Целое число	Вид РК (дубль)
DUE_ADRESAT	Строка (248)	Код Дьюи ДЛ адресата
SEND_DATE	Дата и время	Дата ознакомления
NOTE	Строка (255)	Примечание
ORDERNUM	Целое число	№ в списке
ISN_ACQUAINTANCE	Целое число	Системный идентификатор
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил

ADDR_CATEGORY_CL

Справочник Категории адресатов:

Таблица Г. 2

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN Категории адресатов
CLASSIF_NAME	Строка (64)	Наименование
WEIGHT	Целое число	Вес
PROTECTED	Целое число	Признак защиты от удаления
DELETED	Целое число	Логическое удаление
NOTE	Строка (255)	Примечание
INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Когда добавил

Имя	Тип	Комментарии
UPD_WHO	Целое число	Кто изменил
UPD_DATE	Дата и время	Когда изменил

ADDRESS

Адреса:

Таблица Г. 3

Имя	Тип	Комментарии
ISN_ADDRESS	Целое число	ISN адреса
ISN_ADDRESS_VID	Целое число	ISN вида адреса
ISN_REGION	Целое число	ISN региона
ISN_OWNER	Целое число	ISN владельца
KIND_OWNER	Целое число	Вид владельца
ZIPCODE	Строка (64)	Индекс
SETTLEMENT	Строка (64)	Населенный пункт
ADDRES	Строка (2000)	Адрес
NOTE	Строка (2000)	Примечание
DELETED	Целое число	Логически удален
ORDERNUM	Целое число	Порядковый номер

ADDRESS_VID_CL

Тип адреса:

Таблица Г. 4

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN записи
CLASSIF_NAME	Строка (64)	Название
WEIGHT	Целое число	Вес
PROTECTED	Целое число	Флаг системной записи
DELETED	Целое число	Флаг лог удаления
NOTE	Строка (2000)	Примечание

APP_HOST

Таблица хостов:

Таблица Г. 5

Имя	Тип	Комментарии
ISN_APP_HOST	Целое число	ISN хоста
DISPLAY_NAME	Строка (64)	Название для показа
SERVICE_URL	Строка (2000)	URL сервиса
ENABLED	Целое число	Разрешение на использование

APP_HOST_ASSIGNED_INSTANCE

Физический экземпляр хоста приложения:

Таблица Г. 6

Имя	Тип	Комментарии
ISN_APP_HOST_VIRTUAL	Целое число	ISN хоста
ISN_APP_HOST_INSTANCE	Целое число	ссылка на APP_HOST_INSTANCE

APP_HOST_CONFIG

Настройки экземпляра хоста:

Таблица Г. 7

Имя	Тип	Комментарии
ISN_APP_HOST_CONFIG	Целое число	ISN записи
ISN_APP_HOST_INSTANCE	Целое число	ссылка на APP_HOST_INSTANCE
BINDING_PORT	Целое число	Порт
BINDING_TYPE	Строка (10)	Ссылка на хост
CREATED_AT	Дата и время	Дата время создания

APP_HOST_CONFIG_DAEMON_TYPE

Конфигурация хоста:

Таблица Г. 8

Имя	Тип	Комментарии
ISN_APP_HOST_CONFIG	Целое число	ссылка на APP_HOST_CONFIG
ISN_DAEMON_TYPE_VERSION	Целое число	ссылка на DAEMON_TYPE_VERSION

APP_HOST_INSTANCE

Физический экземпляр хоста приложения:

Таблица Г. 9

Имя	Тип	Комментарии
ISN_APP_HOST_INSTANCE	Целое число	ISN физического экземпляра хоста приложения
COMPUTER_NAME	Строка (64)	Имя компьютера
INSTANCE_NAME	Строка (100)	Имя экземпляра
NOTE	Строка (255)	Примечание

APP_HOST_PULSE

Активность хоста приложения:

Таблица Г. 10

Имя	Тип	Комментарии
ISN_APP_HOST_CONFIG	Целое число	ISN конфигурации хоста приложения
PULSE	Дата и время	Дата и время отметки активности хоста
STATUS	Целое число	Статус
MESSAGE	Строка (128)	Сообщение
RUN_ID	Строка (128)	Идентификатор запуска процесса

APP_HOST_VIRTUAL

Таблица хостов VIRTUAL:

Таблица Г. 11

Имя	Тип	Комментарии
ISN_APP_HOST_VIRTUAL	Целое число	ISN хоста
DISPLAY_NAME	Строка (64)	Название для показа
USER_ID	Целое число	ID пользователя (от которого будет работать виртуальный хост)

APP_SETTINGS

Конфигурирование ФЗ:

Таблица Г. 12

Имя	Тип	Комментарии
NAMESPACE	Строка (64)	Пространство имен
INSTANCE	Строка (64)	Экземпляр
TYPENAME	Строка (64)	Имя типа
PROPERTY	Строка (255)	Свойство
VALUE	Текст	Значение
DATA_TYPE	Целое число	Тип данных
TIMESTAMP	Дата и время	Дата изменения

APPDELTA

Таблица Г. 13

Имя	Тип	Комментарии
APPLICATION_NAME	Приложение	
DELTA	Дельта	

APPROACH

Таблица Г. 14

Имя	Тип	Комментарии
PROPERTI_NAME	VARCHAR2(24)	
PROPERTI_VALUE	VARCHAR2(64)	
WEIGHT	NUMBER(10)	
WEIGHT_DATE	DATE	

AR_CATEGORY

Категории доп реквизитов в поиске:

Таблица Г. 15

Имя	Тип	Комментарии
ISN_NODE	Целое число	ISN категории
ISN_HIGH_NODE	Целое число	ISN вышестоящей вершины
LAYER	Целое число	Номер уровня

IS_NODE	Целое число	Признак вершины
WEIGHT	Целое число	Вес элемента
DUE	Строка (248)	Код DUE
KIND	Целое число	Вид допреквизитов
CLASSIF_NAME	Строка (255)	Наименование категории
ISN_AR_DESCRIPT	Целое число	ISN доп реквизита
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак логического удаления

AR_CITIZEN_VALUE

Допреквизиты граждан:

Таблица Г. 16

Имя	Тип	Комментарии
ISN_CITIZEN	Целое число	ISN гражданина

AR_CLS

Список справочников для заполнения допреквизитов:

Таблица Г. 17

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN справочника
WEIGHT	Целое число	Вес
CLASSIF_NAME	Строка (64)	Наименование справочника
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак логического удаления
NOTE	Строка (255)	Примечание
CLS_EXPRESSION_DEFAULT	Строка (255)	Шаблон значения справочника

AR_CLS_CONTROL

Контроли отображения допреквизитов выбора из справочников на РК и в поиске:

Таблица Г. 18

Имя	Тип	Комментарии
ISN_CONTROL	Целое число	ISN контроля выбора из справочника

Имя	Тип	Комментарии
ISN_CLS_OBJECT	Целое число	ISN справочника из EOS_OBJECTS
IS_DEFAULT	Целое число	Является контролем по умолчанию
CLS_PROP	Строка (255)	Свойства контроля
SRCH_CLS_PROP	Строка (255)	Свойства контроля поиска
CLS_CONTROL	Строка (64)	Тип контроля PB
SRCH_CLS_CONTROL	Строка (64)	Тип контроля поиска PB
CLS_CONTROL_WEB	Строка (64)	Тип контроля WEB
SRCH_CLS_CONTROL_WEB	Строка (64)	Тип контроля поиска WEB
NOTE	Строка (255)	Примечание

AR_CLS_FIELD

Поля справочников для допреквизитов:

Таблица Г. 19

Имя	Тип	Комментарии
ISN_FIELD	Целое число	ISN описания поля
ISN_CLS_OBJECT	Целое число	ISN справочника из EOS_OBJECTS
FIELD_NAME	Строка (64)	Название поля в БД
UI_NAME	Строка (64)	Название поля для отображения
WEIGHT	Целое число	Вес

AR_DESCRIPT

Описание доп реквизитов:

Таблица Г. 20

Имя	Тип	Комментарии
ISN_AR_DESCRIPT	Целое число	ISN описания доп реквизита
OWNER	Строка (1)	Вид реквизита
UI_NAME	Строка (64)	Наименование реквизита
API_NAME	Строка (24)	Имя для API

Имя	Тип	Комментарии
AR_TYPE	Строка (24)	Тип реквизита
USE_LIST_FLAG	Целое число	Использовать список
IS_MULTILINE	Целое число	Является многострочным
MAX_LEN	Целое число	MAX длина текстового поля
AR_PRECISION	Целое число	Знаков после запятой для атрибута DECIMAL
MAX_VAL	Дробное число (18,4)	Максимальное значение атрибута DECIMAL
MIN_VAL	Дробное число (18,4)	Минимальное значение атрибута DECIMAL
DEF_VAL	Строка (255)	Значение по умолчанию
FRM_STR	Строка (255)	Формат представления
ISN_CLS_OBJECT	Целое число	ISN справочника из EOS_OBJECTS
ISN_CLS_CONTROL	Целое число	ISN контроля выбора значения из справочника
CLS_EXPRESSION	Строка (255)	Шаблон значения справочника
UI_READONLY	Целое число	Защищено от редактирования пользователем
NO_COPY	Целое число	Не копировать значение при создании связанной РК или РКПД
ADD_PROT_INFO	Целое число	Создавать доп инфо о протоколе
SEND_ENABLED	Целое число	Отправлять по электронным каналам связи

AR_DOCGROUP

Описание доп реквизита в конкретной гр док и его визуализация:

Таблица Г. 21

Имя	Тип	Комментарии
ISN_AR_DOCGROUP	Целое число	ISN описания доп реквизита в группе документов
ISN_DOCGROUP	Целое число	ISN группы документов

Имя	Тип	Комментарии
ISN_AR_DESCRIPT	Целое число	ISN описания доп реквизита
DEF_VALUE	Строка (2000)	Значение доп реквизита по умолчанию
AR_MANDATORY	Целое число	Обязательность реквизита
TAB_ORDER	Целое число	Последовательность обхода
RX_POS	Целое число	X реквизита на форме
RY_POS	Целое число	Y реквизита на форме
R_HEIGHT	Целое число	Высота реквизита на форме
R_WIDTH	Целое число	Ширина реквизита на форме
R_UOD	Строка (24)	Имя UserObject реквизита
LX_POS	Целое число	X метки на форме
LY_POS	Целое число	Y метки на форме
L_HEIGHT	Целое число	Высота метки на форме
L_WIDTH	Целое число	Ширина метки на форме
L_UOD	Строка (24)	Имя UserObject метки

AR_ORGANIZ_VALUE

Значения доп реквизитов организаций:

Таблица Г. 22

Имя	Тип	Комментарии
ISN_NODE	Целое число	ISN организации

AR_PRJ_VALUE

Значения допреквизитов РКПД:

Таблица Г. 23

Имя	Тип	Комментарии
ISN_PRJ	Целое число	ISN РКПД

AR_RC_VALUE

Значения доп реквизитов РК:

Таблица Г. 24

Имя	Тип	Комментарии
-----	-----	-------------

ISN_RC	Целое число	ISN PK
RC_KIND	Целое число	Вид PK

AR_RUBRIC_VALUE

Дополнительные реквизиты рубрик:

Таблица Г. 25

Имя	Тип	Комментарии
ISN_REF_RUBRIC	Целое число	Ссылка на запись в IREF_Rubric

AR_VALUE_LIST

Допустимые значения полей:

Таблица Г. 26

Имя	Тип	Комментарии
ISN_AR_VALUE_LIST	Целое число	ISN допустимого значения доп. реквизита
ISN_AR_DESCRIPTOR	Целое число	ISN описания доп реквизита
VALUE	Строка (2000)	Значение текстового поля
WEIGHT	Целое число	Вес

BANK_RECVISIT

Банковские реквизиты организации:

Таблица Г. 27

Имя	Тип	Комментарии
ISN_BANK_RECV	Целое число	ISN банковских реквизитов организации
ISN_ORGANIZ	Целое число	Организация
CLASSIF_NAME	Строка (64)	Наименование
BANK_NAME	Строка (64)	Наименование Банка
ACOUNT	Строка (24)	Расчетный счет
SUBACOUNT	Строка (24)	Кор счет
BIK	Строка (9)	БИК
CITY	Строка (64)	Город
NOTE	Строка (255)	Примечание

BAR_CODE_SUPPORT

Счетчик почтовых отправлений:

Таблица Г. 28

Имя	Тип	Комментарии
ISN_BAR_CODE_SUPPORT	Целое число	ISN записи
POST_INDEX	Строка (12)	Почтовый индекс
FIRST_DAY	Дата и время	Первый день
CURR_PACKAGE	Целое число	Текущий номер пакета
MONTH_NUMBER	Целое число	Месяц
MIN_PACKAGE	Целое число	Минимальный номер пакета
MAX_PACKAGE	Целое число	Максимальный номер пакета
POST_OFFICE	Строка (255)	Почтовый офис
UPD_DATE	Дата и время	Время изменения
UPD_WHO	Целое число	Кто изменил

BPM_COMP_PROCESS_TYPE

Типы процесса компонентов:

Таблица Г. 29

Имя	Тип	Комментарии
ISN_COMPONENT_TYPE	Целое число	ISN типа компонентта
ISN_PROCESS_TYPE	Целое число	ISN типа процесса

BPM_COMPONENT_TYPE

Типы компонентов бизнес процессов:

Таблица Г. 30

Имя	Тип	Комментарии
ISN_COMPONENT_TYPE	Целое число	ISN типа компонентта
NAME	Строка (100)	Название
ORDERNUM	Целое число	Порядковый номер
DEFAULT_COMP_NAME	Строка (100)	Название по умолчанию
DEFAULT_COMP_DESC	Строка (2000)	Описание по умолчанию
LEFT_PANEL_AVAILABLE	Целое число	Доступна левая панель

CTX_MENU_AVAILABLE	Целое число	Доступно меню СТХ
UPD_DATE	Дата и время	Дата изменения
UPD_WHO	Целое число	Кто изменил
DELETED	Целое число	Удален

BPM_INSTANCE

Экземпляры процессов:

Таблица Г. 31

Имя	Тип	Комментарии
ISN_INSTANCE	Целое число	Идентификатор
ID_BPMN_INSTANCE	Строка (255)	Идентификатор процесса BPMN Уникальный
ISN_PROCESS_VERSION	Целое число	Ссылка на Версию процесса
KIND_OBJ	Целое число	Код типа
ISN_OBJ	Целое число	Объект владелец
STATE	Целое число	Состояние экземпляра процесса
CREATION_DATE	Дата и время	Дата создания
FINISH_DATE	Дата и время	Дата окончания
ERROR_CODE	Целое число	Код ошибки
ERROR_MESSAGE	Строка (2000)	Текст ошибки

BPM_INSTANCE_COMP_ENDORSE

Параметры компонента Визирование экземпляра процесса:

Таблица Г. 32

Имя	Тип	Комментарии
ISN_INSTANCE_COMP_PARAM	Целое число	Идентификатор
ONLY_ONE	Целое число	ОдинИз
WITH_SUBORDINATES	Целое число	С подчиненными подразделениями
ALLOW_FILE_CHANGE	Целое число	Разрешено изменение файлов проекта
AUTO_COMPLETE_ENDORSEMENT	Целое число	Автоматическое завершение визирования

BPM_INSTANCE_COMP_EXEC

Исполнители параметров компонентов экземпляра процесса:

Таблица Г. 33

Имя	Тип	Комментарии
ISN_INSTANCE_COMP_EXEC	Целое число	Идентификатор исполнителя
ISN_INSTANCE_COMP_PARAM	Целое число	Параметр экземпляра процесса
ISN_EXEC	Целое число	Исполнитель
ISN_ROLE_EXEC	Целое число	Исполнитель роль
ORDERNUM	Целое число	Порядковый номер

BPM_INSTANCE_COMP_PARAM

Параметры компонентов экземпляра процесса:

Таблица Г. 34

Имя	Тип	Комментарии
ISN_INSTANCE_COMP_PARAM	Целое число	Идентификатор
ISN_INSTANCE_COMPONENT	Целое число	Ссылка на Компоненты экземпляра процесса
ORDERNUM	Целое число	Порядок
TERM	Целое число	Срок
TERM_TYPE	Целое число	Тип срока По умолчанию рабочие дни
NOTE	Строка (2000)	Комментарий
APPLY_EDS	Целое число	Применять ЭП По умолчанию Из группы документов

BPM_INSTANCE_COMP_PREP

Параметры компонента Подготовка экземпляра процесса:

Таблица Г. 35

Имя	Тип	Комментарии
ISN_INSTANCE_COMP_PARAM	Целое число	Идентификатор
ONLY_ONE	Целое число	ОдинИз
WITH_SUBORDINATES	Целое число	С подчиненными подразделениями
CAN_READ_PRJ_RC	Целое число	Чтение проекта документа

Имя	Тип	Комментарии
CAN_WORK_WITH_PRJ	Целое число	Работа с проектом документа
CAN_WORK_WITH_FILES	Целое число	Работа с файлами проекта документа
AUTO_COMPLETE_PREPARATION	Целое число	Автоматическое завершение подготовки

BPM_INSTANCE_COMP_SIGN

Параметры компонента Подписание экземпляра процесса:

Таблица Г. 36

Имя	Тип	Комментарии
ISN_INSTANCE_COMP_PARAM	Целое число	Идентификатор
ALLOW_FILE_CHANGE	Целое число	Разрешено изменение файлов проекта

BPM_INSTANCE_COMPONENT

Компоненты экземпляра процесса:

Таблица Г. 37

Имя	Тип	Комментарии
ISN_INSTANCE_COMPONENT	Целое число	Идентификатор
ISN_INSTANCE	Целое число	Ссылка на экземпляр
ISN_PROC_VER_COMPONENT	Целое число	Ссылка на компонент версии процесса
ID_BPMN_COMPONENT	Строка (255)	Идентификатор компонента BPMN
STATE	Целое число	Состояние компонента экземпляра процесса
ACTIVATION_DATE	Дата и время	Дата активации
FINISH_DATE	Дата и время	Дата окончания
ERROR_CODE	Целое число	Код ошибки
ERROR_MESSAGE	Строка (2000)	Текст ошибки
ISN_COMPONENT_TYPE	Целое число	Тип компонента
CAN_CHANGE_PARAMS	Целое число	Разрешено изменение параметров
ASSIGNMENT_TYPE	Целое число	Тип направления

BPM_PROC_VER_COMP_ENDORSE

Параметры компонента Визирование версии процесса:
Таблица Г. 38

Имя	Тип	Комментарии
ISN_PROC_VER_COMP_PARAM	Целое число	Идентификатор
ONLY_ONE	Целое число	ОдинИз
WITH_SUBORDINATES	Целое число	С подчиненными подразделениями
ALLOW_FILE_CHANGE	Целое число	Разрешено изменение файлов проекта

BPM_PROC_VER_COMP_EXEC

Исполнители параметров компонентов версии процесса:
Таблица Г. 39

Имя	Тип	Комментарии
ISN_PROC_VER_COMP_EXEC	Целое число	Идентификатор исполнителя
ISN_PROC_VER_COMP_PARAM	Целое число	Параметр версии процесса
ISN_EXEC	Целое число	Исполнитель
ISN_ROLE_EXEC	Целое число	Исполнитель роль
DELETED	Целое число	Удален
ORDERNUM	Целое число	Порядковый номер

BPM_PROC_VER_COMP_PARAM

Параметры компонентов версии процесса:
Таблица Г. 40

Имя	Тип	Комментарии
ISN_PROC_VER_COMP_PARAM	Целое число	Идентификатор
ISN_PROC_VER_COMPONENT	Целое число	Ссылка на Компоненты процесса
ISN_OBJ	Целое число	Объект
KIND_OBJ	Целое число	Тип объекта
ORDERNUM	Целое число	Порядковый номер
TERM	Целое число	Срок
TERM_TYPE	Целое число	Тип срока

Имя	Тип	Комментарии
NOTE	Строка (2000)	Комментарий
APPLY_EDS	Целое число	Применять ЭП

BPM_PROC_VER_COMP_PREP

Параметры компонента Подготовка версии процесса:
Таблица Г. 41

Имя	Тип	Комментарии
ISN_PROC_VER_COMP_PARAM	Целое число	Идентификатор
ONLY_ONE	Целое число	ОдинИз
WITH_SUBORDINATES	Целое число	С подчиненными подразделениями
CAN_READ_PRJ_RC	Целое число	Чтение проекта документа Значение по умолчанию = Временно
CAN_WORK_WITH_PRJ	Целое число	Работа с проектом документа
CAN_WORK_WITH_FILES	Целое число	Работа с файлами проекта документа Значение по умолчанию = Временно

BPM_PROC_VER_COMP_SIGN

Параметры компонента Подписание версии процесса:
Таблица Г. 42

Имя	Тип	Комментарии
ISN_PROC_VER_COMP_PARAM	Целое число	Идентификатор
ALLOW_FILE_CHANGE	Целое число	Разрешено изменение файлов проекта

BPM_PROC_VER_COMPONENT

Компоненты версии процесса:
Таблица Г. 43

Имя	Тип	Комментарии
ISN_PROC_VER_COMPONENT	Целое число	Идентификатор
ID_BPMN_COMPONENT	Строка (255)	Идентификатор компонента BPMN
ISN_COMPONENT_TYPE	Целое число	Тип компонента
ISN_PROCESS_VERSION	Целое число	ISN версии процесса

Имя	Тип	Комментарии
CAN_CHANGE_PARAMS	Целое число	Разрешено изменение параметров
ASSIGNMENT_TYPE	Целое число	Тип направления

BPM_PROC_VER_DG

Для связи версии процесса и группы документов:

Таблица Г. 44

Имя	Тип	Комментарии
ISN_PROCESS_VERSION	Целое число	ISN версии процесса
DUE_DOCGROUP	Строка (248)	DUE группы документов

BPM_PROC_VER_OBJECT

Объекты версии процесса:

Таблица Г. 45

Имя	Тип	Комментарии
ISN_PROC_VER_OBJECT	Целое число	Идентификатор
ISN_PROCESS_VERSION	Целое число	Ссылка на версию процесса
ISN_OBJ	Целое число	Объект
KIND_OBJ	Целое число	Тип объекта

BPM_PROC_VER_PRJ

Реквизиты процесса типа Ведение проекта документа:

Таблица Г. 46

Имя	Тип	Комментарии
ISN_PROCESS_VERSION	Целое число	Идентификатор версии
START_TERM_ON	Целое число	Начало отсчета срока
FINISH_TERM_ON	Целое число	Вид переноса окончания срока

BPM_PROC_VER_PRJ_STAGE

Для связи версии процесса и значения статусов:

Таблица Г. 47

Имя	Тип	Комментарии
ISN_PROCESS_VERSION	Целое число	ISN версии процесса
ISN_PRJ_STAGE	Целое число	ISN статуса РКПД

BPM_PROC_VER_SCHEDULE

Реквизиты Расписания:

Таблица Г. 48

Имя	Тип	Комментарии
ISN_PROC_VER_SCHEDULE	Целое число	Идентификатор
ISN_PROCESS_VERSION	Целое число	Ссылка на версию процесса
SCHEDULE_TYPE	Целое число	Тип расписания
SCHEDULE	Строка (100)	Расписание

BPM_PROCESS

Процессы:

Таблица Г. 49

Имя	Тип	Комментарии
ISN_PROCESS	Целое число	ISN процесса
ISN_PROCESS_TYPE	Целое число	ISN типа процесса
DELETED	Целое число	Удален
IS_SYSTEM	Целое число	Системный

BPM_PROCESS_ROLE

Роли в бизнес процессах:

Таблица Г. 50

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	Идентификатор
CLASSIF_NAME	Строка (255)	Название
NOTE	Строка (2000)	Описание
WEIGHT	Целое число	Порядок
ISN_PERSON	Целое число	Должностное лицо
PROTECTED	Целое число	Защищенный
ROLE_TYPE	Целое число	Тип роли
INS_DATE	Дата и время	Дата создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата изменения

Имя	Тип	Комментарии
UPD_WHO	Целое число	Кто изменил
DELETED	Целое число	Удален

BPM_PROCESS_TYPE

Типы процессов:

Таблица Г. 51

Имя	Тип	Комментарии
ISN_PROCESS_TYPE	Целое число	ISN типа процесса
NAME	Строка (2000)	Название
ORDERNUM	Целое число	Порядковый номер
DESCRIPTION	Строка (2000)	Описание
UPD_DATE	Дата и время	Дата изменения
UPD_WHO	Целое число	Кто изменил
DELETED	Целое число	Удален

BPM_PROCESS_VERSION

Версии процессов:

Таблица Г. 52

Имя	Тип	Комментарии
ISN_PROCESS_VERSION	Целое число	ISN версии процесса
ISN_PROCESS	Целое число	ISN процесса
STATUS	Целое число	Статус
VERSION	Целое число	Версия
CURRENT_FLAG	Целое число	Последняя версия
COMPONENT_SCHEMA	Текст	Последовательность компонентов
ACTIVE_FROM	Дата и время	Дата и время начала работы
ACTIVE_TO	Дата и время	Дата и время окончания работы
MANUAL_START	Целое число	Разрешение на ручной запуск
EVENT_START	Целое число	Запуск по событию

Имя	Тип	Комментарии
SCHEDULE_START	Целое число	Запуск по расписанию
INS_DATE	Дата и время	Дата создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата изменения
UPD_WHO	Целое число	Кто изменил
DELETED	Целое число	Удален
DEL_DATE	Дата и время	Дата удаления
DEL_WHO	Целое число	Кто удалил
PUBLISH_DATE	Дата и время	Дата публикации
LOCK_FLAG	Целое число	Блокировка
ISN_USER_LOCK	Целое число	Кто заблокировал
LOCK_DATE	Дата и время	Дата блокировки
NAME	Строка (100)	Название
DESCRIPTION	Строка (2000)	Описание

BPM_USER_PROCESS_TYPE

Типы процессов по пользователям:

Таблица Г. 53

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN пользователя
ISN_PROCESS_TYPE	Целое число	ISN типа процесса

BUF_FOLDER

Папка сообщение буфера:

Таблица Г. 54

Имя	Тип	Комментарии
ISN_FOLDER	Целое число	ISN папки
NAME	Строка (255)	Название
WEIGHT	Целое число	Вес
KIND_FOLDER	Целое число	Вид папки

Имя	Тип	Комментарии
ADDRESS	Строка (255)	Элект адрес отправителя
TEXT_BODY	Строка (4000)	Текст сообщения

BUF_MESSAGE

Сообщение буфера:

Таблица Г. 55

Имя	Тип	Комментарии
ISN_MESSAGE	Целое число	ISN сообщения
IMPORTANCE	Целое число	Важность сообщения
SENDER	Строка (255)	От кого поступило сообщение
RECIPIENT	Строка (255)	Кому поступило сообщение и кому поступила копия
SUBJECT	Строка (2000)	Тема сообщения
RECEIVE_DATE	Дата и время	Дата поступления сообщения
BODY	Текст	Текст сообщения
READ_MESSAGE	Целое число	Прочитано или не прочитано сообщение
ISN_FOLDER	Целое число	ISN папки в которой лежит сообщение
ENTRYID	Строка (255)	Идентификатор электронного сообщения в источнике
SENDER_ADDR	Строка (255)	Электронный адрес отправителя
RECIPIENT_ADDR	Строка (255)	Электронный адрес получателя
STATUS_MESSAGE	Целое число	Текущий статус сообщения
DELETED	Целое число	Признак логического удаления сообщения
NOTE	Строка (2000)	Примечание
FACT_ADDR	Строка (255)	Значение фактического электронного адреса на почтовом сервере
REPLY_FROM_ADDR	Строка (255)	Эл адрес с которого должны отправляться ответные сообщения

Имя	Тип	Комментарии
TYPE_MESSAGE	Целое число	Тип сообщения
SOURCE	Строка (255)	Источник
EDS_FLAG	Целое число	Флаг ЭД
ENCRYPT_FLAG	Целое число	Флаг шифрования
RECIPIENT_CERTIFICATE	Текст	Сертификат получателя
DELETED_DATE	Дата и время	Дата установки флага удаления
UMESSAGEID	Строка (255)	Составной идентификатор входящего почтового сообщения
SEND_DATE	Дата и время	Дата отправки
EPVV_FINAL_ADDRESS	Строка (255)	Адрес конечного получателя
EDS_RESULT	Строка (2000)	Результат

BUF_MESSAGE_FIELDS

Поля сообщения буфера:

Таблица Г. 56

Имя	Тип	Комментарии
ISN_MESSAGE	Целое число	ISN сообщения
CORRESP_NUM	Строка (64)	Номер документа у корреспондента
CORRESP_DATE	Дата и время	Дата документа у корреспондента
PASSPORT_ADDRESSEE	Текст	Список адресатов из паспорта системы «ДЕЛО»
GUID_DOC	Строка (255)	GUID документа
MEDO_FORMAT	Строка (64)	Формат МЭДО
MEDO_SOURCE	Строка (512)	Источник МЭДО
MEDO_KIND	Строка (512)	Вид МЭДО
MEDO_ADDRESSEE	Текст	Адресат МЭДО
MEDO_CHANNEL	Строка (64)	Канал МЭДО
MEDO_GATE_ID	Строка (255)	ID организации МЭДО

Имя	Тип	Комментарии
REPEATED	Строка (255)	Повторность

BUF_RECEIPT_COUNT

Счетчик квитанций буфера:

Таблица Г. 57

Имя	Тип	Комментарии
COUNT_BASE	Строка (64)	Владелец
PERIOD	Дата и время	Период
COUNT_NAME	Строка (64)	Название счетчика
COUNT_VALUE	Целое число	Значение счетчика

BUF_RULE

Правила буфера:

Таблица Г. 58

Имя	Тип	Комментарии
ISN_BUF_RULE	Целое число	ISN правила
ISN_FOLDER	Целое число	ISN папки
DUE_DEP	Строка (248)	DUE узла справочника подразделений
SUBJECT	Строка (255)	Тема
INCLUDE_MEMBERS	Целое число	Включая членов
SOURCE	Текст	Адрес получателя
RESTRICT_CHANNEL	Строка (64)	RESTRICT_CHANNEL

BUF_USER_FOLDER

Папки пользователя буфера:

Таблица Г. 59

Имя	Тип	Комментарии
ISN_FOLDER	Целое число	ISN папки
ISN_USER	Целое число	ISN пользователя
ISDEFAULT	Целое число	По умолчанию

BULK_DELETE_PROT

Протокол операции зачистки справочников:

Таблица Г. 60

Имя	Тип	Комментарии
TABLE_ID	Целое число	Идентификатор таблицы
REF_ISN	Целое число	ISN записи справочника
CLASSIF_NAME	Строка (255)	Наименование справочника
DELETED	Целое число	Статус удаления
IS_NODE	Целое число	Признак Листа
NOTE	Строка (255)	Примечание

CA_CATEGORY

Группы УЦ:

Таблица Г. 61

Имя	Тип	Комментарии
ISN_CA_CATEGORY	Целое число	ISN записи
CA_SERIAL	Текст	Серийный номер корневого сертификата УЦ
CA_SUBJECT	Текст	Тема корневого сертификата УЦ
ISN_EDS_CATEGORY	Целое число	Категория ЭП

CABINET

Кабинеты:

Таблица Г. 62

Имя	Тип	Комментарии
ISN_CABINET	Целое число	ISN кабинета
DUE	Строка (248)	Код подразделения
CABINET_NAME	Строка (64)	Имя кабинета
FULLNAME	Строка (2000)	Полное наименование

CALENDAR_CL

Календарь:

Таблица Г. 63

Имя	Тип	Комментарии
ISN_CALENDAR	Целое число	ISN календаря

Имя	Тип	Комментарии
DATE_CALENDAR	Дата и время	Дата
DATE_TYPE	Целое число	Тип даты

CB_PRINT_INFO

Печатные формы:

Таблица Г. 64

Имя	Тип	Комментарии
ISN_OWNER	Целое число	ISN владельца
OWNER_KIND	Целое число	Вид владельца
PRINT_SURNAME	Строка (64)	ФИО
PRINT_SURNAME_DP	Строка (64)	ФИО в дательном
PRINT_SURNAME_RP	Строка (64)	ФИО в родительном
PRINT_DUTY	Строка (255)	Должность
PRINT_DEPARTMENT	Строка (255)	Подразделение
DEPARTMENT_RP	Строка (255)	Подразделение (родительный падеж)
NOT_USE_IN_DUTY	Целое число	Признак использования подразделения в названии должности
SURNAME	Строка (64)	Фамилия (именительный падеж)
NAME	Строка (64)	Имя (именительный падеж)
PATRON	Строка (64)	Отчество (именительный падеж)
SURNAME_RP	Строка (64)	Фамилия (родительный падеж)
NAME_RP	Строка (64)	Имя (родительный падеж)
PATRON_RP	Строка (64)	Отчество (родительный падеж)
SURNAME_DP	Строка (64)	Фамилия (дательный падеж)
NAME_DP	Строка (64)	Имя (дательный падеж)
PATRON_DP	Строка (64)	Отчество (дательный падеж)
SURNAME_VP	Строка (64)	Фамилия (винительный падеж)

NAME_VP	Строка (64)	Имя (винительный падеж)
PATRON_VP	Строка (64)	Отчество (винительный падеж)
SURNAME_TP	Строка (64)	Фамилия (творительный падеж)
NAME_TP	Строка (64)	Имя (творительный падеж)
PATRON_TP	Строка (64)	Отчество (творительный падеж)
SURNAME_PP	Строка (64)	Фамилия (предложный падеж)
NAME_PP	Строка (64)	Имя (предложный падеж)
PATRON_PP	Строка (64)	Отчество (предложный падеж)
GENDER	Целое число	Пол
DUTY_RP	Строка (255)	Должность (родительный падеж)
DUTY_DP	Строка (255)	Должность (дательный падеж)
DUTY_VP	Строка (255)	Должность (винительный падеж)

CERTIFICATE

Сертификаты:

Таблица Г. 65

Имя	Тип	Комментарии
ISN_CERTIFICATE	Целое число	ISN сертификата
ID_CERTIFICATE	Строка (4000)	Идентификатор сертификата
CERT_KIND	Целое число	Тип сертификата
DESCRIPTION	Строка (2000)	Описание
CERT_BODY	Большой двоичный объект	Тело сертификата

CITIZEN

Справочник Граждане:

Таблица Г. 66

Имя	Тип	Комментарии
ISN_CITIZEN	Целое число	ISN гражданина
CITIZEN_SURNAME	Строка (64)	Фамилия И О

Имя	Тип	Комментарии
CITIZEN_SURNAME_SEARCH	Строка (64)	Фамилия И О в верхн регистре
CITIZEN_CITY	Строка (255)	Город
ISN_REGION	Целое число	Регион
CITIZEN_CITY_SEARCH	Строка (255)	Город в верхн регистре
ZIPCODE	Строка (12)	Почтовый индекс
CITIZEN_ADDR	Строка (255)	Адрес
DELETED	Целое число	Признак логического удаления
PROTECTED	Целое число	Признак запрета удаления
WEIGHT	Целое число	Вес элемента
ISN_ADDR_CATEGORY	Целое число	Категория адресата
PHONE	Строка (64)	Телефон
SEX	Целое число	Пол
N_PASPORT	Строка (64)	N Паспорта
SERIES	Строка (64)	Серия
GIVEN	Строка (255)	Выдан
INN	Строка (64)	ИНН
NEW	Целое число	Признак новой записи
E_MAIL	Строка (255)	e_mail
EDS_FLAG	Целое число	Требуется ЭП
ENCRYPT_FLAG	Целое число	Требуется шифрование
ID_CERTIFICATE	Строка (4000)	Идентификатор сертификата
NOTE	Строка (255)	Комментарий
MAIL_FORMAT	Целое число	Почтовый формат
SNILS	Строка (14)	Снилс
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал

Имя	Тип	Комментарии
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил

CITIZEN_STATUS

Статусы гражданм:

Таблица Г. 67

Имя	Тип	Комментарии
ISN_CIT_STAT	Целое число	идентификатор
ISN_STATUS	Целое число	ISN статуса
ISN_CITIZEN	Целое число	ISN гражданина

CITSTATUS_CL

Справочник статусов граждан:

Таблица Г. 68

Имя	Тип	Комментарии
DUE	Строка (248)	Код Дьюи статуса
ISN_NODE	Целое число	ISN статуса
ISN_HIGH_NODE	Целое число	№ верхней вершины
LAYER	Целое число	Номер уровня
IS_NODE	Целое число	Признак вершины
WEIGHT	Целое число	Вес элемента
MAXDUE	Строка (248)	MAX значение кода Дьюи
CLASSIF_NAME	Строка (64)	Наименование статуса
PROTECTED	Целое число	Пр запрета удаления
DELETED	Целое число	Пр лог удаления
CODE	Строка (64)	Код
NOTE	Строка (255)	Комментарий
INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Когда добавил
UPD_WHO	Целое число	Кто изменил

Имя	Тип	Комментарии
UPD_DATE	Дата и время	Когда изменил

CL_SEARCH

Тексты для поиска по справочникам:

Таблица Г. 69

Имя	Тип	Комментарии
ISN_CL_SEARCH	Целое число	ISN записи
ISN_OWNER	Целое число	ISN записи справочника
KIND_OWNER	Целое число	Вид справочника
SEARCH_TEXT	Текст	Поисковый образ записи

CONNECTION_LOG

Журнал коннектов пользователей:

Таблица Г. 70

Имя	Тип	Комментарии
WORKSTATION	Строка (255)	Рабочая станция
APPLICATION	Строка (64)	Приложение
DELO_USER	Строка (64)	Логин пользоавателя Дело
OS_USER	Строка (255)	Имя пользователя windows
VERSION	Строка (255)	Версия приложения
DATE_TIME	Дата и время	Дата время коннекта
USER_STATISTICS	Текст	Статистика по пользователям

CONTACT

Контакты в организации:

Таблица Г. 71

Имя	Тип	Комментарии
ISN_CONTACT	Целое число	ISN контакта
ID_CERTIFICATE	Строка (4000)	Идентификатор сертификата
ISN_ORGANIZ	Целое число	ISN организации
ORDERNUM	Целое число	№ в списке
DELETED	Целое число	Признак лог удаления

Имя	Тип	Комментарии
SURNAME	Строка (64)	ФИО
SURNAME_DP	Строка (64)	ФИО в дательном
DUTY	Строка (255)	Должность
DEPARTMENT	Строка (255)	Подразделение
PHONE_LOCAL	Строка (24)	Телефон местный
PHONE	Строка (24)	Телефон городской
FAX	Строка (24)	Факс
E_MAIL	Строка (255)	e_mail
EDS_FLAG	Целое число	Требуется ЭП
ENCRYPT_FLAG	Целое число	Требуется шифрование
MAIL_FORMAT	Целое число	Почтовый формат
NOTE	Строка (2000)	Примечание
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил
SEV_INDEX	Строка (255)	Индекс СЭВ
MEDO_GLOBAL_ID	Строка (127)	Глобальный идентификатор организации
MEDO_ID	Строка (127)	Идентификатор организации

CUSTOM_STORAGE

Спецхранилище значений:

Таблица Г. 72

Имя	Тип	Комментарии
VALUE_ID	Строка (255)	Идентификатор значения
ORDERNUM	Целое число	Порядковый номер
VALUE	Строка (2000)	Значение
OWNER_KIND	Целое число	Тип объекта владельца

Имя	Тип	Комментарии
OWNER_ID	Строка (255)	Идентификатор объекта владельца

DAEMON_GROUP

Группа фоновых задач:

Таблица Г. 73

Имя	Тип	Комментарии
ISN_DAEMON_GROUP	Целое число	ISN группы фоновых задач
NAME	Строка (255)	Название

DAEMON_INSTANCE

Экземпляр фоновой задачи:

Таблица Г. 74

Имя	Тип	Комментарии
ISN_DAEMON_INSTANCE	Целое число	ISN экземпляра фоновой задачи
ISN_DAEMON_TYPE	Целое число	ISN типа фоновой задачи
ISN_APP_HOST_VIRTUAL	Целое число	ISN виртуального экземпляра хоста приложения
DAEMON_INSTANCE_ID	Строка (128)	Идентификатор экземпляра фоновой задачи
ALLOWED	Целое число	Признак разрешения работы экземпляра
DISPLAY_NAME	Строка (255)	Отображаемое имя экземпляра

DAEMON_INSTANCE_BIND

Привязка экземпляров фоновых задач:

Таблица Г. 75

Имя	Тип	Комментарии
ISN_APP_HOST_INSTANCE	Целое число	Ссылка на APP_HOST_INSTANCE
ISN_DAEMON_INSTANCE	Целое число	Ссылка на DAEMON_INSTANCE
LAST_DATE	Дата и время	Дата выполнения задачи
TASK_ID	Строка (255)	Идентификатор экземпляра ФЗ на самом хосте

DAEMON_TYPE

Тип фоновой задачи:

Таблица Г. 76

Имя	Тип	Комментарии
ISN_DAEMON_TYPE	Целое число	ISN типа фоновой задачи
ISN_DAEMON_GROUP	Целое число	ISN группы фоновых задач
CLASS_NAME	Строка (128)	Полное имя
LIBRARY_NAME	Строка (128)	Имя библиотеки
DAEMON_NAME	Строка (128)	Имя фоновой задачи

DAEMON_TYPE_VERSION

Версии библиотек фоновых задач:

Таблица Г. 77

Имя	Тип	Комментарии
ISN_DAEMON_TYPE_VERSION	Целое число	ISN записи
ISN_DAEMON_TYPE	Целое число	ссылка на DAEMON_TYPE
LIBRARY_VERSION	Строка (64)	Версия библиотеки
MULTI_INSTANCE	Целое число	Поддержка нескольких экземпляров
FLAGS	Строка (255)	Перечень флагов
KIND_OBJECTS	Строка (64)	Перечень типов объектов
DAEMON_KIND	Целое число	Тип фоновой задачи

DELIVERY_CL

Справочник видов доставки:

Таблица Г. 78

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN вида доставки
WEIGHT	Целое число	Вес элемента
CLASSIF_NAME	Строка (64)	Наименование вида
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак лог удаления
NOTE	Строка (255)	Комментарий

Имя	Тип	Комментарии
E_SENDING_FLAG	Целое число	Флаг электронной отправки
INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Когда добавил
UPD_WHO	Целое число	Кто изменил
UPD_DATE	Дата и время	Когда изменил

DELO_BLOB

Файлы объектов Дело:

Таблица Г. 79

Имя	Тип	Комментарии
ISN_BLOB	Целое число	ISN объекта
CONTENTS	Большой двоичный объект	Содержимое
EXTENSION	Строка (64)	Расширение

DELO_OWNER

Организация владелец системы:

Таблица Г. 80

Имя	Тип	Комментарии
ISN_DELO_OWNER	Целое число	ISN записи
NAME	Строка (255)	Название Организации
ISN_ORGANIZ	Целое число	Организация
ORG_ID	Строка (64)	Идентификационный номер организации
SYS_ID	Строка (64)	GUID системы
SYS_NAME	Строка (64)	= «ДЕЛО»
SYS_VERSION	Строка (64)	Номер версии системы

DEP_REPLACE

Замещающие ДЛ:

Таблица Г. 81

Имя	Тип	Комментарии
DUE	Строка (248)	DUE ДЛ

START_DATE	Дата и время	Начальная дата
END_DATE	Дата и время	Конечная дата
REASON	Строка (64)	Причина замещения
DUE_REPLACE	Строка (248)	DUE замещающего
HISTORY	Строка (2000)	История замещения

DEPARTMENT

Справочник подразделений (ДЛ):

Таблица Г. 82

Имя	Тип	Комментарии
DUE	Строка (248)	Код Дьюи ДЛ
ISN_NODE	Целое число	ISN ДЛ
ISN_ORGANIZ	Целое число	Не используется Организация
ISN_HIGH_NODE	Целое число	Номер вышестоящей вершины
LAYER	Целое число	Номер уровня
IS_NODE	Целое число	Признак вершины
WEIGHT	Целое число	Вес элемента
MAXDUE	Строка (248)	MAX значение кода Дьюи
CLASSIF_NAME	Строка (255)	Наименование ДЛ
SURNAME	Строка (64)	Фамилия
DUTY	Строка (255)	Должность
FULLNAME	Строка (2000)	Полное наименование
CODE	Строка (64)	Код
SKYPE	Строка (64)	Skype
DEPARTMENT_DUE	Строка (248)	Картотека ДЛ
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак логического удаления
ISN_CABINET	Целое число	ISN кабинета
ORDER_NUM	Целое число	Порядковый номер в кабинете

Имя	Тип	Комментарии
DEPARTMENT_INDEX	Строка (24)	Индекс ДЛ
POST_H	Целое число	Признак начальника
CARD_FLAG	Целое число	Признак образования картотеки
CARD_NAME	Строка (64)	Наименование картотеки
NOTE	Строка (255)	Комментарий
START_DATE	Дата и время	Дата начала действия
END_DATE	Дата и время	Дата окончания действия
ISN_CONTACT	Целое число	ISN контакта
PHONE_LOCAL	Строка (24)	№ местн тел
PHONE	Строка (24)	№ тел
FAX	Строка (24)	Факс
E_MAIL	Строка (64)	E_MAIL
NUM_CAB	Строка (24)	№ каб
DUE_LINK_ORGANIZ	Строка (248)	Дие связанной организации
ISN_PHOTO	Целое число	ISN фотографии
EXPEDITION_FLAG	Целое число	Флаг отправки
NUMCREATION_FLAG	Целое число	Флаг номерообразования
ID_GAS_PS	Строка (10)	ID в системе ГАС
MEDO_ID	Строка (127)	ID МЭДО
SYNC_ID	Строка (255)	Идентификатор из кадров НСИ
UNREAD_FLAG	Целое число	Запись не проверена технологом
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил

DEPARTMENT_REPL

Делегирование полномочий ДЛ:

Таблица Г. 83

Имя	Тип	Комментарии
ISN_DEPARTMENT_REPL	Целое число	ISN делегирования полномочий ДЛ
DUE_DEP	Строка (248)	Кого замещаем
DUE_REPL	Строка (248)	Кем замещаем
START_DATE	Дата и время	Дата начала действия
END_DATE	Дата и время	Дата окончания действия

DG_FILE_CATEGORY

Категории файлов групп документов:

Таблица Г. 84

Имя	Тип	Комментарии
ISN_NODE_DG	Целое число	Группа документов
ISN_FILE_CATEGORY	Целое число	Категория файлов

DG_FILE_CONSTRAINT

Ограничения на файлы для групп документов:

Таблица Г. 85

Имя	Тип	Комментарии
ISN_DOCGROUP	Целое число	ISN группы
CATEGORY	Строка (64)	Категория
MAX_SIZE	Целое число	Максимальный размер
ONE_FILE	Целое число	Для одного файла
EXTENSIONS	Строка (255)	Расширения

DIADOC_EXCHANGE

Диадок обмен:

Таблица Г. 86

Имя	Тип	Комментарии
ISN_DIADOC_EXCHANGE	Целое число	ISN обмена Диадок
OBJECT_ID	Строка (255)	ID объекта

OBJECT_NAME	Строка (64)	Название объекта
GLOBAL_ID	Строка (255)	Глобальный ID

DIADOC_SUBSCRIPTION

Подписки Диадок:

Таблица Г. 87

Имя	Тип	Комментарии
ISN_DIADOC_SUBSCRIPTION	Целое число	ISN подписки Диадок
ISN_SUBSCRIBE	Целое число	ISN файла, подпись которого ожидается
ISN_DOC	Целое число	ISN созданной РКПД
KIND_DOC	Целое число	Вид документа
VISA_SIGN_UIDS	Текст	GUID направления на подпись
MESSAGE_ID	Строка (64)	ID входящего сообщения из диадока
ENTITY_ID	Строка (64)	ID файла документа из диадока
BOX_ID	Строка (64)	ID ящика отправителя из диадока
HASH	Строка (255)	хэш доклада

DOC_DEFAULT

Список идентификаторов реквизитов РК и их описание:

Таблица Г. 88

Имя	Тип	Комментарии
DEFAULT_ID	Строка (255)	Идентификатор реквизита
DEFAULT_TYPE	Строка (1)	Тип реквизита
KIND_DOC	Строка (24)	Тип РК
DESCRIPTION	Строка (255)	Описание реквизита
CLASSIF_ID	Целое число	ссылки реквизита

DOC_DEFAULT_VALUE

Правила заполнения реквизитов РК:

Таблица Г. 89

Имя	Тип	Комментарии
-----	-----	-------------

ISN_DOCGROUP	Целое число	ISN группы документов
DEFAULT_ID	Строка (255)	Идентификатор реквизита
VALUE	Строка (2000)	Значение реквизита

DOC_EXE

Исполнители РК:

Таблица Г. 90

Имя	Тип	Комментарии
ISN_DOC_EXE	Целое число	ISN записи об исполнителе
ISN_DOC	Целое число	ISN документа
KIND_DOC	Целое число	Вид документа
DUE_PERSON	Строка (248)	DUE исполнителя
ORDERNUM	Целое число	Порядковый номер
INS_DATE	Дата и время	Когда добавили
INS_WHO	Целое число	Кто добавил
UPD_DATE	Дата и время	Когда обновили
UPD_WHO	Целое число	Кто обновил

DOC_FOLDER_ITEM

Документы в папках кабинетов:

Таблица Г. 91

Имя	Тип	Комментарии
ISN_FOLDER_ITEM	Целое число	ISN элемента папки
ISN_FOLDER	Целое число	ISN папки
ISN_DOC	Целое число	ISN документа
ISN_RESOLUTION	Целое число	ISN резолюции
ISN_REPLY	Целое число	ISN ответа
GREEN_FLAG	Целое число	"Нетронутый элемент папки"
REMINDER	Целое число	"Наличие напоминания"
INS_DATE	Дата и время	время создания
INS_WHO	Целое число	создатель

Имя	Тип	Комментарии
HOLD	Целое число	Оставлять ли ссылку на РК при пересчете
WITHRESOLUTION	Целое число	флаг Рассмотрено
BOSS_VIEW	Целое число	Запись видна руководителю

DOC_ORGANIZ_EXCLUDE

Исключение документов из картотек организаций:

Таблица Г. 92

Имя	Тип	Комментарии
ISN_DOC	Целое число	ISN документа
DUE_LINK_ORGANIZ	Строка (248)	DUE организации

DOC_RC

РК документов:

Таблица Г. 93

Имя	Тип	Комментарии
ISN_DOC	Целое число	ISN документа
KIND_DOC	Целое число	Вид РК
DUE_DOCGROUP	Строка (248)	Группа документов
DOC_DATE	Дата и время	Дата регистрации
DOC_YEAR	Целое число	Год регистрации
ORDER_NUM	Целое число	№ в группе
FREE_NUM	Строка (64)	Рег №
FREE_NUM_SEARCH	Строка (64)	Рег № верх регистр
ANNOTAT	Строка (2000)	Краткое содержание
CONSISTS	Строка (255)	Состав документа
SPECIMEN	Строка (64)	Экземпляльность
ISN_CARD_REG	Целое число	Картотека регистрации
ISN_CABINET_REG	Целое число	Кабинет регистратора
ISN_NOMENC	Целое число	Дело

Имя	Тип	Комментарии
SECURLEVEL	Целое число	Гриф доступа
ISN_DELIVERY	Целое число	Вид доставки
ISREPEAT	Целое число	Признак повторности
ISCOLLECTIVE	Целое число	Признак коллективности
ADRESS_FLAG	Целое число	Тип РК
TEL_NUM	Строка (64)	Почтовый номер
ANONIM	Целое число	анонимное
DUE_PERSON_EXE	Строка (248)	Кто исполнил
CONTROL_STATE	Целое число	На контроле
PLAN_DATE	Дата и время	Плановая дата
FACT_DATE	Дата и время	Фактическая дата
DELTA_DATE	Целое число	Нарушение срока
NOTE	Строка (2000)	Комментарий
FILE_COUNT	Целое число	Количество файлов
FILE_COUNT_VISIBLE	Целое число	Количество видимых файлов
NOTHARDCOPY	Целое число	Без досылки бум экз
CITO	Целое число	Срочно
E_DOCUMENT	Целое число	Признак электронного документа
FREE_NUM_SORT	Строка (255)	Номер для сортировки
OTHER_WHO	Целое число	РК была изменена не регистратором
ACCESS_MODE	Целое число	Конфиденциальность
SENDER_NUM	Строка (64)	Рег номер корреспондента
INS_DATE	Дата и время	Когда создал
INS_WHO	Целое число	Кто создал запись
EDIT_REASON	Строка (255)	Причина редактирования РК

Имя	Тип	Комментарии
ISN_DOCVID	Целое число	Вид документа

DOC_SIGN

Подписи документа:

Таблица Г. 94

Имя	Тип	Комментарии
ISN_DOC_SIGN	Целое число	ISN подписи
ISN_DOC	Целое число	ISN документа
KIND_DOC	Целое число	Вид РК
DUE_PERSON	Строка (248)	DUE должностного лица
SIGN_DATE	Дата и время	Дата подписания
ORDERNUM	Целое число	№ в списке
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил
SIGN_NUM	Строка (64)	РК исх док внеш ДЛ

DOC_TEMPLATES

Шаблоны печатных форм:

Таблица Г. 95

Имя	Тип	Комментарии
ISN_TEMPLATE	Целое число	ISN шаблона
KIND_TEMPLATE	Целое число	Вид шаблона
NAME_TEMPLATE	Строка (64)	Наименование
DESCRIPTION	Строка (255)	Описание
INFO	Строка (255)	Структура шаблона
DELETED	Целое число	Пр лог удаления
PROTECTED	Целое число	Пр запрета удаления
WEIGHT	Целое число	Вес элемента

CATEGORY	Строка (64)	Категория
FILECONTENTS	Большой двоичный объект	Содержимое файла
INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Когда добавил
UPD_WHO	Целое число	Кто изменил
UPD_DATE	Дата и время	Когда изменил

DOC_WHO

"Кому" входящего документа или письма:

Таблица Г. 96

Имя	Тип	Комментарии
ISN_DOC_WHO	Целое число	ISN получателя
ISN_DOC	Целое число	ISN документа
KIND_DOC	Целое число	Вид РК
DUE_PERSON	Строка (248)	DUE должностного лица
ORDERNUM	Целое число	№ в списке
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил

DOCGROUP_CL

Справочник Группы документов:

Таблица Г. 97

Имя	Тип	Комментарии
DUE	Строка (248)	Код Дьюи группы
ISN_NODE	Целое число	ISN группы
ISN_HIGH_NODE	Целое число	Номер вышестоящей вершины
LAYER	Целое число	Номер уровня
IS_NODE	Целое число	Признак вершины
IS_COPYCOUNT	Целое число	Признак нумерации копий

Имя	Тип	Комментарии
WEIGHT	Целое число	Вес элемента
MAXDUE	Строка (248)	MAX значение кода Дьюи
CLASSIF_NAME	Строка (64)	Наименование группы
FULLNAME	Строка (2000)	Полное наименование
CODE	Строка (64)	Код
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак лог удаления
RC_TYPE	Целое число	Вид РК
DOCGROUP_INDEX	Строка (24)	Индекс группы
DOCNUMBER_FLAG	Целое число	Признак образования номеров
SHABLON	Строка (64)	Шаблон номера документа
EDS_FLAG	Целое число	Требуется ЭП
ENCRYPT_FLAG	Целое число	Требуется шифрование
TEST_UNIQ_FLAG	Целое число	Проверять уникальность
PRJ_NUM_FLAG	Целое число	Разрешение создания РКПД
PRJ_SHABLON	Строка (64)	Шаблон номера проекта документа
PRJ_WEIGHT	Целое число	Вес в списке для проектов
PRJ_AUTO_REG	Целое число	Авторегистрация проекта
PRJ_APPLY_EDS	Целое число	Применять ЭП для подписания проекта
PRJ_APPLY2_EDS	Целое число	Применять ЭП для визирования проекта
PRJ_APPLY_EXEC_EDS	Целое число	Применять ЭП для исполнителя проекта
PRJ_DEL_AFTER_REG	Целое число	Удалять РК проекта после регистрации
PRJ_TEST_UNIQ_FLAG	Целое число	Флаг проверки уникальности номера проекта
E_DOCUMENT	Целое число	Электронный документ

Имя	Тип	Комментарии
NOTE	Строка (255)	Комментарий
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил
ACCESS_MODE	Целое число	Конфиденциальность
INITIATIVE_RESOLUTION	Целое число	флаг Инициативная резолюция
PROTECT_DEL_PRJ_STATUS	Целое число	Запрет удаления РПКП со статуса
COPY_NUMBER_FLAG	Целое число	Копировать номер
COPY_NUMBER_FLAG_PRJ	Целое число	Копировать номер РПКД
ACCESS_MODE_FIXED	Целое число	Без редактирования
REG_DATE_PROTECTED	Целое число	Запрет редактирования даты регистрации
ISN_DOCVID	Целое число	Вид документа

DOCVID_CL

Виды документов:

Таблица Г. 98

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN записи
CLASSIF_NAME	Строка (64)	Наименовани
FULLNAME	Строка (2000)	Полное наименование
DOCVID_INDEX	Строка (64)	Индекс вида
AP_FLAG	Целое число	Флаг отправки в Аппарат Правительства
WEIGHT	Целое число	Вес
PROTECTED	Целое число	Признак защищенной записи
DELETED	Целое число	Призна логического удаления
NOTE	Строка (255)	Примечание

INS_WHO	Целое число	Кто создал
INS_DATE	Дата и время	Когда создал
UPD_WHO	Целое число	Кто изменил
UPD_DATE	Дата и время	Когда изменил

EDS_CATEGORY_CL

Категории ЭП:

Таблица Г. 99

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN записи
CLASSIF_NAME	Строка (64)	Название
WEIGHT	Целое число	Вес
PROTECTED	Целое число	Защита от изменения
DELETED	Целое число	Защита от удаления
NOTE	Строка (255)	Примечание
INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Когда добавил
UPD_WHO	Целое число	Кто изменил
UPD_DATE	Дата и время	Когда изменил

EOS_JOB

Задания СУБД:

Таблица Г. 100

Имя	Тип	Комментарии
JOB_NAME	Строка (255)	Задание
LAST_RUN_DATE	Дата и время	Дата время последнего исполнения
LAST_RUN_STATUS	Строка (255)	Статус последнего выполнения
LAST_ERR_DATE	Дата и время	Дата и время последней ошибки
LAST_ERR_MESSAGE	Строка (255)	Сообщение последней ошибки
STATUS	Целое число	Состояние задания

EOS_SCN

Отметка номера структурного изменения БД:

Таблица Г. 101

Имя	Тип	Комментарии
SCN	Целое число	Номер изменения
SCN_VER	Целое число	Версия изменения

EVNT_FEED

События для потребления:

Таблица Г. 102

Имя	Тип	Комментарии
ISL_EVENT	Целое число	ISL события
KIND_EVENT	Целое число	вид события
KIND_OBJECT	Целое число	вид объекта
OBJECT_ID	Строка (255)	ID объекта
DUE_DOCGROUP	Строка (248)	DUE группы документов
FLAGS	Строка (2000)	флаги
TIME_STAMP	Дата и время	Метка времени
DATA_I1	Целое число	Данные I1
DATA_S1	Строка (2000)	Данные S1
DATA_D1	Дата и время	Данные D1
DATA_I2	Целое число	Данные I2
DATA_S2	Строка (2000)	Данные S2
DATA_D2	Дата и время	Данные D2
DATA_I3	Целое число	Данные I3
DATA_S3	Строка (2000)	Данные S3
DATA_D3	Дата и время	Данные D3
DATA_I4	Целое число	Данные I4
DATA_S4	Строка (2000)	Данные S4
DATA_D4	Дата и время	Данные D4

Имя	Тип	Комментарии
DATA_I5	Целое число	Данные I5
DATA_S5	Строка (2000)	Данные S5
DATA_D5	Дата и время	Данные D5

EVNT_OBSERVED

Наблюдение за событиями:

Таблица Г. 103

Имя	Тип	Комментарии
OBJECT_ID	Строка (255)	ID объекта
ISN_REQUEST	Целое число	ISN запроса
EXPIRED	Дата и время	Дата истечения
DATA	Текст	Данные

EVNT_QUEUE_ITEM

Элементы очереди подписки:

Таблица Г. 104

Имя	Тип	Комментарии
ISN_SUBSCRIPTION	Целое число	ISN подписчика
ISN_EVENT	Целое число	ISN события
ISL_EVENT	Целое число	ISL события
KIND_EVENT	Целое число	Вид события
KIND_OBJECT	Целое число	Вид объекта
OBJECT_ID	Строка (255)	Идентификатор объекта
DUE_DOCGROUP	Строка (248)	Две группы документов
FLAGS	Строка (2000)	Флаги события
DATA_I1	Целое число	Данные I1
DATA_S1	Строка (2000)	Данные S1
DATA_D1	Дата и время	Данные D1
DATA_I2	Целое число	Данные I2
DATA_S2	Строка (2000)	Данные S2

DATA_D2	Дата и время	Данные D2
DATA_I3	Целое число	Данные I3
DATA_S3	Текст	Данные S3
DATA_D3	Дата и время	Данные D3
DATA_I4	Целое число	Данные I4
DATA_S4	Текст	Данные S4
DATA_D4	Дата и время	Данные D4
DATA_I5	Целое число	Данные I5
DATA_S5	Текст	Данные S5
DATA_D5	Дата и время	Данные D5

EVNT_SUBSCRIPTION

Подписка на получение сообщений:

Таблица Г. 105

Имя	Тип	Комментарии
ISN_SUBSCRIPTION	Целое число	ISN подписчика
ID_APPLICATION	Строка (255)	Идентификатор приложения
APPLICATION_NAME	Строка (255)	Имя приложения
MAX_EVENT_COUNT	Целое число	Максимальное количество обработанных записей

EXT_PROT

Внешнее протоколирование:

Таблица Г. 106

Имя	Тип	Комментарии
ISN_EXT_PROT	Целое число	ISN внешнего протоколирования
DATE_TIME	Дата и время	Дата внешнего протоколирования
MACHINE	Строка (255)	Имя компьютера
USER_ID	Строка (64)	Логин пользователя
USER_NAME	Строка (64)	Имя пользователя
PROCESS	Строка (64)	Имя приложения

KIND_OBJECT	Строка (255)	Вид объекта
ISN_OBJECT	Целое число	ISN объекта
KIND_PARENT_OBJECT	Строка (255)	Вид родительского объекта
ISN_PARENT_OBJECT	Целое число	ISN родительского объекта
OPER_NAME	Строка (64)	Имя операции
OPER_RESULT	Строка (24)	Результат операции
RESULT_COMMENT	Строка (255)	Комментарий к результату

EXT_PROT_OPER_TYPES

Типы операций внешнего протоколирования:

Таблица Г. 107

Имя	Тип	Комментарии
ISN_OPER_TYPE	Целое число	ISN типа операции
OPER_NAME	Строка (24)	Имя операции

EXT_PROT_SETTINGS

Настройка внешнего протоколирования:

Таблица Г. 108

Имя	Тип	Комментарии
REFRESH_PERIOD	Целое число	Период обновления
MAX_PROT_COUNT	Целое число	Максимальное количество
IS_ACTIVE	Целое число	Признак "Протоколирование включено"
MAX_RECORDS_TO_FILE	Целое число	Максимальное количество записей на файл

FILE_CATEGORY_CL

Категории файлов:

Таблица Г. 109

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	Идентификатор
NAME	Строка (64)	Название
NOTE	Строка (2000)	Примечание
WEIGHT	Целое число	Порядок

PROTECTED	Целое число	Флаг системного элемента
DELETED	Целое число	Флаг лог удаления

FILE_CONTENTS

Содержимое файлов документа (режим хранения DB):

Таблица Г. 110

Имя	Тип	Комментарии
ISN_REF_FILE	Целое число	ISN файла
ORDER_NUMBER	Целое число	№ куска файла
COMPR_FLAG	Целое число	Признак компрессии
CONTENTS	Большой двоичный объект	Содержимое файла

FILE_CONTENTS2

Содержимое файлов документа (режим хранения DB2):

Таблица Г. 111

Имя	Тип	Комментарии
ISN_REF_FILE	Целое число	ISN файла
FILE_EXTENSION	Строка (255)	Расширение файла
CONTENTS	Большой двоичный объект	Содержимое файла
REG_DATE	Дата и время	Дата регистрации объекта

FILE_TYPE_CL

Типы файлов:

Таблица Г. 112

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN типов файлов
CLASSIF_NAME	Строка (64)	Название
DELETED	Целое число	Логическое удаление
PROTECTED	Целое число	Признак защиты от удаления
WEIGHT	Целое число	Вес
TAG	Строка (64)	ТЭГ
UNIQUE_FLAG	Целое число	Флаг уникальности
NOTE	Строка (255)	Примечание

INS_DATE	Дата и время	Дата создания
INS_WHO	Целое число	Кто добавил
UPD_DATE	Дата и время	Дата изменения
UPD_WHO	Целое число	Кто изменил

FOLDER

Папки кабинетов:

Таблица Г. 113

Имя	Тип	Комментарии
ISN_FOLDER	Целое число	ISN папки
ISN_CABINET	Целое число	ISN кабинета
FOLDER_KIND	Целое число	Тип папки
USER_COUNT	Целое число	"Папка открыта для наполнения"

FORMAT_CL

Настройки для поточного сканирования:

Таблица Г. 114

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN настройки для поточного сканирования
FORMAT_TNAME	Строка (24)	Текстовый формат
KIND_ADJ	Целое число	Не используется
FILE_ADJ	Строка (255)	Не используется
FORMAT_GNAME	Строка (24)	Графический формат
COLOR	Целое число	Цветовая гамма
COMPR	Целое число	Метод сжатия
WEIGHT	Целое число	Вес
PROTECTED	Целое число	Запрет удаления
DEL_COL	Целое число	Признак логического удаления
NOTE	Строка (64)	Наименование
PRIORITET	Целое число	По умолчанию

FORUM_DUE

Обсуждение адресованные ДЛ:

Таблица Г. 115

Имя	Тип	Комментарии
ISN_FORUM_THEME	Целое число	ISN темы
DUE_DEP	Строка (248)	DUE ДЛ

FORUM_MSG

Сообщения обсуждения:

Таблица Г. 116

Имя	Тип	Комментарии
ISN_FORUM_MSG	Целое число	ISN сообщения
ISN_FORUM_THEME	Целое число	Тема обсуждения
MSG_TEXT	Строка (2000)	Текст сообщения
MSG_TYPE	Целое число	Тип сообщения
MSG_QUOTED	Целое число	ISN цитируемого сообщения
INS_WHO	Целое число	Кто отправил сообщение
INS_DATE	Дата и время	Дата создания сообщения
UPD_DATE	Дата и время	Дата изменения сообщения

FORUM_READ

Обсуждение количество прочитанных сообщений:

Таблица Г. 117

Имя	Тип	Комментарии
ISN_FORUM_THEME	Целое число	ISN темы
ISN_USER	Целое число	ISN пользователя
READ_MSG_COUNT	Целое число	Количество прочитанных сообщений в теме

FORUM_THEME

Темы обсуждения:

Таблица Г. 118

Имя	Тип	Комментарии
ISN_FORUM_THEME	Целое число	ISN темы

Имя	Тип	Комментарии
THEME_NAME	Строка (255)	Название темы
ISN_OWNER	Целое число	ISN владельца
KIND_OWNER	Целое число	Вид владельца
IS_COMMON	Целое число	Общая
MSG_COUNT	Целое число	Кол сообщений в теме
INS_WHO	Целое число	Кто создал тему
INS_DATE	Дата и время	Когда создали тему
MSG_UPD_WHO	Целое число	Кто написал последнее сообщение
MSG_UPD_DATE	Дата и время	Когда написано последнее сообщение

FORWARD

Журнал пересылок РК:

Таблица Г. 119

Имя	Тип	Комментарии
ISN_FORWARD	Целое число	Системный идентификатор
ISN_DOC	Целое число	ISN документа
KIND_DOC	Целое число	Вид документа
ISN_USER	Целое число	isn пользователя
DUE_CORRESP	Строка (248)	Код Дьюи картотеки корресп
ISN_CABINET	Целое число	ISN кабинета корреспондента
SEND_DATE	Дата и время	Дата пересылки
SEND_CARD_ONLY	Целое число	Рассылать только в картотеку не в кабинеты
DUE_ADRESAT	Строка (248)	Код Дьюи ДЛ адресата

FUZZY_WORD

Словарь слов нечеткого поиска:

Таблица Г. 120

Имя	Тип	Комментарии
ISN_WORD	Целое число	ISN слова

WORD	Строка (128)	Слово
------	--------------	-------

FUZZY_WORD_OBJ

Связки словаря нечеткого поиска:

Таблица Г. 121

Имя	Тип	Комментарии
ISN_WORD	Целое число	ISN слова
KIND_OBJECT	Целое число	Вид объекта
ISN_OBJ	Целое число	ISN объекта
COUNT	Целое число	Количество

HELPER_T1

Вспомогательная таблица:

Таблица Г. 122

Имя	Тип	Комментарии
ISN	Целое число	ISN объекта
DUE	Строка (248)	DUE объекта

JOURNAL

Журнал передачи документов:

Таблица Г. 123

Имя	Тип	Комментарии
ISN_JOURNAL	Целое число	Системный идентификатор
KIND_DOC	Целое число	Вид документа
ISN_DOC	Целое число	ISN документа
DUE_CORRESP	Строка (248)	Код Дьюи картотеки корресп
ISN_CABINET	Целое число	ISN кабинета корреспондента
ISN_NOMENC	Целое число	Дело
DUE_ADRESAT	Строка (248)	Код Дьюи ДЛ адресата
SEND_DATE	Дата и время	Дата передачи
REESTR_DATE	Дата и время	Дата получения реестра
REESTR_NUM	Целое число	Номер реестра
NOTES	Строка (255)	Примечания

Имя	Тип	Комментарии
ORIGINAL_NUM	Целое число	Номер оригинала
COPY_NAME	Строка (24)	Номер копии
ORIG_FLAG	Целое число	Флаг копии
OWNER_FLAG	Целое число	Флаг владельца
REC_TYPE	Целое число	Тип записи
VOL_NUM	Целое число	Номер тома
VOL_STR	Целое число	Страниц в томе
STR_DOC	Целое число	Листов документа
EXPORT_DATE	Дата и время	Дата выгрузки
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил

LIB_FILELINK

Ссылка на файл:

Таблица Г. 124

Имя	Тип	Комментарии
ISN_FILELINK	Целое число	ISN ссылки
ISN_LIBRARY	Целое число	ISN библиотеки
PATH	Строка (512)	Путь
ID	Строка (255)	Идентификатор
FILE_EXTENSION	Строка (24)	Расширение файла

LIB_LIBRARY

Библиотека хранения:

Таблица Г. 125

Имя	Тип	Комментарии
ISN_LIBRARY	Целое число	ISN библиотеки
NAME	Строка (64)	Название

DESCRIPTION	Строка (255)	Описание
-------------	--------------	----------

LIB_PARAM

Параметры библиотеки хранения:

Таблица Г. 126

Имя	Тип	Комментарии
ISN_LIB_PARAM	Целое число	ISN записи
ISN_LIBRARY	Целое число	ISN библиотеки
PARAM_NAME	Строка (255)	Название параметра
PARAM_VALUE	Строка (2000)	Значение параметра

LINK_CL

Справочник Типы связей:

Таблица Г. 127

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN типа связи
WEIGHT	Целое число	Вес элемента
CLASSIF_NAME	Строка (64)	Имя типа связи
LINK_TYPE	Целое число	Тип связки
LINK_DIR	Целое число	Направленность связки
PROTECTED	Целое число	Пр запрета удаления
DELETED	Целое число	Пр лог удаления
TRANSPARENT	Целое число	Есть доступ к связанному объекту
ISN_PARE_LINK	Целое число	ISN парной связи
LINK_INDEX	Строка (24)	Индекс связки
NOTE	Строка (255)	Примечание
INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Когда добавил
UPD_WHO	Целое число	Кто изменил
UPD_DATE	Дата и время	Когда изменил
MEDO_ID	Строка (127)	Идентификатор МЭДО

LIST_ITEMS

Элементы списка пользователя:

Таблица Г. 128

Имя	Тип	Комментарии
ISN_LIST	Целое число	ISN списка пользователя
REF_ISN	Целое число	ISN записи справочника
KIND_OBJ	Целое число	Тип справочника
WEIGHT	Целое число	Вес элемента
NOTE	Строка (255)	Примечание

MAIL_FOLDER

Почтовые папки:

Таблица Г. 129

Имя	Тип	Комментарии
ISN_FOLDER	NUMBER(10)	ISN папки
NAME	VARCHAR2(255)	Наименование
WEIGHT	NUMBER(10)	Вес папки

MAIL_RULE

Почтовые правила:

Таблица Г. 130

Имя	Тип	Комментарии
ISN_MAIL_RULE	NUMBER(10)	ISN почтового правила
ISN_FOLDER	NUMBER(10)	ISN папки
DUE_DEP	VARCHAR2(48)	Код Дьюи
SUBJECT	VARCHAR2(255)	Содержание
INCLUDE_MEMBERS	NUMBER (1)	Кто входит в группу

MEDO_DISPATCHER_PROT

Журнал работы диспетчера МЭДО:

Таблица Г. 131

Имя	Тип	Комментарии
ISN_MEDO_DISPATCHER_PROT	Целое число	ISN записи
ID_MSG	Строка (36)	ID сообщения

Имя	Тип	Комментарии
ID_DOC	Строка (36)	ID документа
DOC_TYPE	Целое число	Тип
FORMAT	Строка (10)	Номер формата
MSG_DIRECTION	Целое число	Направление сообщения
IS_DSP	Целое число	Признак ДСП документа
RESULT_STATUS	Целое число	Результат операции
GATE_CONTACT_DATE	Дата и время	Дата время контакта со шлюзом
NODE_CONTACT_DATE	Дата и время	Дата время контакта с узлом
SENDER_GATE_ID	Строка (128)	Оператор отправитель идентификатор шлюза
SENDER_ORGANIZ_ID	Строка (128)	Организация отправитель идентификатор
SENDER_ORGANIZ_NAME	Строка (512)	Организация отправитель наименование
RECEIVER_ORGANIZ_ID	Строка (128)	Организация получатель идентификатор
RECEIVER_ORGANIZ_NAME	Строка (512)	Организация получатель наименование
LOCAL_FOLDER	Строка (512)	Адрес физической папки
PROT_COMMENT	Строка (512)	Комментарий
ISN_MEDO_NODE	Целое число	Узел из в который перемещено получено сообщение

MEDO_NODE_CL

Узлы МЭДО:

Таблица Г. 132

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN записи
NAME	Строка (255)	Название узла
DIRECTORY	Строка (255)	Каталог обмена
PASSWORD	Строка (64)	Пароль

NOTE	Строка (2000)	Примечание
WEIGHT	Целое число	Вес
DELETED	Целое число	Флаг лог удаление
PROTECTED	Целое число	Флаг защищенной записи
INS_DATE	Дата и время	Дата создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата изменения
UPD_WHO	Целое число	Кто изменил

MEDO_PARTICIPANT

Участники МЭДО:

Таблица Г. 133

Имя	Тип	Комментарии
ISN_ORGANIZ	Целое число	ISN организации
GATE	Строка (255)	Адрес МЭДО
GATE_ID	Строка (255)	Идентификатор шлюза
MSGFORMAT	Строка (64)	Формат обмена
MEDO_SUBFOLDER_IN	Строка (512)	Папка МЭДО входная
MEDO_SUBFOLDER_OUT	Строка (512)	Папка МЭДО выходная
GATE_OWNER	Целое число	Признак владельца шлюза
ISN_REAL_ORGANIZ	Целое число	ISN реальной организации
ISN_MAIN_NODE	Целое число	Узел общих сообщений
ISN_DSP_NODE	Целое число	Узел ДСП сообщений

MESSAGES

Таблица Г. 134

Имя	Тип	Комментарии
LANGUAGE	VARCHAR2(3)	language
MSGID	VARCHAR2(64)	msgid
MSGTITLE	VARCHAR2(255)	msgtitle
MSGTEXT	VARCHAR2(255)	msgtext

Имя	Тип	Комментарии
MSGICON	VARCHAR2(24)	msgicon
MSGBUTTON	VARCHAR2(24)	msgbutton
MSGDEFAULTBUTTON	NUMBER(4)	msgdefaultbutton
MSGSEVERITY	NUMBER(10)	MSGSEVERITY
MSGPRINT	CHAR(1)	msgprint
MSGUSERINPUT	CHAR(1)	msguserinput

MODULES

Таблица Г. 135

Имя	Тип	Комментарии
VERSION	NUMBER(10)	Номер версии
APP_PART	VARCHAR2(24)	Приложение
MOD_NAME	VARCHAR2(24)	Наименование модуля
PART_NUMBER	NUMBER(10)	Номер части модуля
PART_TOTAL	NUMBER(10)	Количество частей
PART_CONTENTS	BLOB	Тело части модуля

MREVT_ASSOCIATION

ДЛ и подразделения в событиях календаря:

Таблица Г. 136

Имя	Тип	Комментарии
ISN_ASSOCIATION	Целое число	Идентификатор записи
ISN_EVENT	Целое число	Ссылка на событие
DUE_DEPARTMENT	Строка (248)	Ссылка на запись в справочнике Подразделения

MREVT_EVENT

События календаря:

Таблица Г. 137

Имя	Тип	Комментарии
ISN_EVENT	Целое число	Идентификатор записи

Имя	Тип	Комментарии
DUE_DEPARTMENT	Строка (248)	Ссылка на ДЛ в справочнике Подразделения
EVENT_TYPE	Целое число	Тип события в календаре
TITLE	Строка (255)	Тема/название пункта плана
BODY	Строка (2000)	Текст заметки / Примечание к пункту плана
EVENT_DATE	Дата и время	Дата/время, на которое создана заметка.
EVENT_DATE_TO	Дата и время	Дата/время окончания события.
EVENT_DATE_TO_FACT	Дата и время	Фактическая дата/время окончания события
EVENT_DATE_FACT	Дата и время	Фактическая дата/время начала события.
PLACE	Строка (2000)	Место проведения события.
REASON	Строка (2000)	Основание проведения события.
IS_PERSONAL	Целое число	Признак принадлежности события.
IS_COMMON	Целое число	Признак «С вложенными подразделениями».
STATUS	Целое число	Статус события.
RESULTS	Строка (2000)	Причины отмены события
ISN_MTG_MEETING	Целое число	Ссылка на событие, созданное на основании пункта плана
INS_DATE	Дата и время	Дата/время создания записи.
INS_WHO	Целое число	Кем создана.
UPD_DATE	Дата и время	Дата/время обновления записи.
UPD_WHO	Целое число	Кем обновлена.

MREVT_REF

Ссылки в событиях календаря:

Таблица Г. 138

Имя	Тип	Комментарии
-----	-----	-------------

Имя	Тип	Комментарии
ISN_REF	Целое число	Идентификатор записи
ISN_EVENT	Целое число	Ссылка на события
REF_KIND	Строка (64)	Тип объекта
ISN_DOC	Целое число	Ссылка на документ
URL	Строка (2048)	URL-адрес

MTG_AR_MEETING_VALUE

Дополнительные реквизиты события:

Таблица Г. 139

Имя	Тип	Комментарии
ISN_MTG_MEETING	Целое число	ISN события
SES_CONVOCAATION	Строка (64)	Созыв сессии
SES_NUMBER	Строка (64)	Номер сессии
SES_TYPE	Целое число	Тип сессии
SES_BEG_DATE	Дата и время	Дата начала сессии
SES_END_DATE	Дата и время	Дата окончания сессии

MTG_DECISION_TYPE

Типы решения

Таблица Г. 140

Имя	Тип	Комментарии
ISN_MTG_DECISION_TYPE	Целое число	ISN типа решения
NAME	Строка (255)	Наименование

MTG_INVITED

Ответственные:

Таблица Г. 141

Имя	Тип	Комментарии
ISN_MTG_INVITED	Целое число	ISN ответственного
ISN_MTG_ISSUE	Целое число	ISN вопроса
ISN_MTG_PARTICIPANT	Целое число	ISN участника

MTG_ISSUE_RUBRIC

Рубрики вопросов:

Таблица Г. 142

Имя	Тип	Комментарии
ISN_ISSUE_RUBRIC	Целое число	ISN рубрики вопроса
ISN_MTG_ISSUE	Целое число	ISN вопроса
DUE_RUBRIC	Строка (248)	DUE рубрики

MTG_ISSUE_TYPE

Типы вопросов:

Таблица Г. 143

Имя	Тип	Комментарии
ISN_MTG_ISSUE_TYPE	Целое число	ISN типа вопроса
NAME	Строка (255)	Наименование

MTG_MEETING_RUBRIC

Рубрики событий:

Таблица Г. 144

Имя	Тип	Комментарии
ISN_MEETING_RUBRIC	Целое число	ISN рубрики события
ISN_MTG_MEETING	Целое число	ISN события
DUE_RUBRIC	Строка (248)	DUE рубрики

NEW_RECORD_CABINET

В кабинете пользователя есть новые записи:

Таблица Г. 145

Имя	Тип	Комментарии
ISN_USER	Целое число	ISN пользователя
ISN_CABINET	Целое число	ISN кабинета

NOMENKL_CL

Справочник Номенклатура дел:

Таблица Г. 146

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN дела
DUE	Строка (248)	Код Дьюи подразделения

Имя	Тип	Комментарии
WEIGHT	Целое число	Вес элемента
CLASSIF_NAME	Строка (2000)	Наименование дела
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак лог удаления
CLOSED	Целое число	Флаг активности дела
SECURITY	Строка (64)	Гриф доступа
YEAR_NUMBER	Целое число	Год создания записи
STORE_TIME	Целое число	Срок хранения
SHELF_LIFE	Строка (255)	Срок хранения дела
NOM_NUMBER	Строка (24)	Индекс дела
NOM_NUMBER_SORT	Строка (255)	Индекс дела для лексикографической сортировки
NOTE	Строка (512)	Комментарий
END_YEAR	Целое число	Год завершения дела
ARTICLE	Строка (255)	Статья
INS_WHO	Целое число	Кто создал
INS_DATE	Дата и время	Когда создана запись
UPD_WHO	Целое число	Кто отредактировал запись
UPD_DATE	Дата и время	Когда отредактировал
CLOSE_WHO	Целое число	CLOSE_WHO
CLOSE_DATE	Дата и время	CLOSE_DATE
DOC_UID	Строка (512)	Системные номера Ланита
ARCH_DATE	Дата и время	Дата передачи в Архив
ARCH_FLAG	Целое число	Подлежит сдаче в архив
E_DOCUMENT	Целое число	флаг для электронный документов

NOTIFY_CL

Классификатор сроков напоминаний:

Таблица Г. 147

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN срока напоминания
CLASSIF_NAME	Строка (64)	Название срока напоминания
PROTECTED	Целое число	Признак защиты от удаления
DELETED	Целое число	Логическое удаление
NOTE	Строка (255)	Примечание
WEIGHT	Целое число	Вес записи
SROK	Целое число	Срок напоминания

NTFY_EVENT

Описатель события прошедшего в системе:

Таблица Г. 148

Имя	Тип	Комментарии
ISN_EVENT	Целое число	ISN описателя события
OPERATION_CODE	Строка (5)	Код операции
TYPE_OBJECT	Строка (5)	Тип сущности
ISN_OBJECT	Целое число	ISN объекта экземпляра
EVENT_TIME	Дата и время	Дата и время проведения операции
ISN_USER	Целое число	ISN пользователя
MAIN_SYS_PARM	Целое число	Параметр основной системы
DUE_PERSON	Строка (248)	Идентификатор должностного лица
STATE	Целое число	Состояние описателя прошедшего события
SERVICE_ID	Строка (248)	Идентификатор системного агента

NTFY_MESSAGE

Тексты сообщений оповещений:

Таблица Г. 149

Имя	Тип	Комментарии
ISN_NTFY_MSG_TEXT	NUMBER(10)	ISN записи
CODE	VARCHAR(5)	Код операции
MSG_MOMENT	NUMBER(4)	Момент оповещения
MSG_TEXT	VARCHAR2(255)	Основной текст оповещения
MSG_ADDTEXT	VARCHAR2(2000)	Дополнение к тексту сообщения
OPERATION_TAGS	VARCHAR2(255)	Список тегов
EXAMPLE_TEXT	VARCHAR2(255)	Пример основного текста
EXAMPLE_ADDTEXT	VARCHAR2(255)	Пример доп текста по умолчанию

NTFY_OPERATION

Операции и шаблоны сообщений, отправляемых пользователю:

Таблица Г. 150

Имя	Тип	Комментарии
CODE	Строка (5)	Код операции
NAME	Строка (128)	Пользовательское наименование операции
KIND	Целое число	Применение
MSG_HEADER	Строка (255)	Шаблон заголовок сообщения
MSG_TEXT	Текст	Текст сообщения
MSG_EXTEND_TEXT	Текст	Дополнительный текст сообщения
MSG_EXTEND_TEXT2	Текст	Дополнительный текст сообщения2
NOTE	Строка (2000)	Примечание

NTFY_PERIODICITY

Периодичность уведомлений:

Таблица Г. 151

Имя	Тип	Комментарии
ISN_PERIODICITY	Целое число	ISN периодичности уведомлений
NAME	Строка (64)	Пользовательское наименование периодичности
KIND	Целое число	Вид периодичности
PARAMETERS	Строка (255)	Параметры расчета графика
NOTE	Строка (2000)	Примечание
DELETED	Целое число	Признак логического удаления

NTFY_SUBSCRIPTION

Подписка на оповещения и уведомления:

Таблица Г. 152

Имя	Тип	Комментарии
ISN_SUBSCRIPTION	Целое число	ISN оповещений и уведомлений
OPERATION_CODE	Строка (5)	Код операции
ISN_PERIODICITY	Целое число	ISN периодичности уведомлений
USER_ROLE	Строка (10)	Список ролей пользователя
IS_ENABLED	Целое число	Доступность
INIT_DATE	Дата и время	Дата инициализации
LAST_PROCESSING_DATE	Дата и время	Дата последней обработки данных по этим параметрам
RESOLUTION_CONTROL_STATE	Целое число	Признак контрольности

NTFY_SYSTEM_PARAMS

Параметры системы оповещений и уведомлений:

Таблица Г. 153

Имя	Тип	Комментарии
SERVICE_ID	Строка (248)	Идентификатор системного агента

SYSTEM_STATE	Целое число	Состояние системы
MAX_EVENT_COUNT	Целое число	Максимальное количество обработанных записей
SERVICE_HOST	Строка (64)	Сетевой адрес компьютера службы
SERVICE_PORT	Целое число	Номер порта службы
SERVICE_NAME	Строка (255)	Название системной службы

NTFY_USER_EMAIL

Параметры пользователей системы оповещений и уведомлений:

Таблица Г. 154

Имя	Тип	Комментарии
ISN_NTFY_USER_EMAIL	Целое число	ISN записи
ISN_USER	Целое число	ISN пользователя
EMAIL	Строка (255)	Адрес электронной почты
IS_ACTIVE	Целое число	Доступность
WEIGHT	Целое число	Вес
EXCLUDE_OPERATION	Строка (2000)	Перечень событий по которым в этот адрес НЕ посылаются сообщения
DUE_DEP	Строка (248)	ДЛ
ISN_ORGANIZ	Целое число	ISN организации
TYPE_CHANNEL	Целое число	Тип канала

NTFYPF_CHANNEL

Каналы отправки:

Таблица Г. 155

Имя	Тип	Комментарии
NTFYPF_CHANNEL_ID	Целое число	ID канала отправки
CHANNEL_SIZE_TYPE	Целое число	Тип размера сообщений канала
CHANNEL_UI_NAME	Строка (64)	Название канала в UI

NTFYPF_MESSAGE

Сообщения уведомлений

Таблица Г. 156

Имя	Тип	Комментарии
NTFYPF_MESSAGE_ID	Целое число	ID сообщения
NTFYPF_CHANNEL_ID	Целое число	ID канала отправки
ADDRESSES	Строка (2000)	Список адресатов сообщения
HIDE_COPY	Целое число	Скрытая копия
MESSAGE	Текст	Текст сообщения
HEADER	Строка (255)	Заголовок сообщения
CITO	Целое число	Важность сообщения
CREATED_AT	Дата и время	Время создания сообщения
OPERATION_CODE	Строка (5)	Код операции

NTFYPF_MESSAGE_JOURNAL

Журнал отправки сообщений:

Таблица Г. 157

Имя	Тип	Комментарии
NTFYPF_MESSAGE_JOURNAL_ID	Целое число	ID Журнала сообщения
NTFYPF_MESSAGE_ID	Целое число	ID сообщения
ERROR_SENDS_COUNT	Целое число	Количество неудачных попыток отправки
LAST_TRY_AT	Дата и время	Последняя попытка отправки
STATUS	Целое число	Статус сообщения

NUMCREATION

Счетчики номеробразования РК:

Таблица Г. 158

Имя	Тип	Комментарии
ISN_DOCGROUP	Целое число	Номер группы
YEAR_NUMBER	Целое число	Год
ISN_NUM_BASE	Целое число	ISN объекта
CURRENT_NUMBER	Целое число	Текущий номер
FLAG_MAX	Целое число	Признак макс номера

USER_ID	Строка (255)	Ид пользователя
LOCK_TIME	Дата и время	Время захвата записи

OBJ_TEMPLATE

Шаблоны объектов:

Таблица Г. 159

Имя	Тип	Комментарии
ISN_OBJ_TEMPLATE	Целое число	ISN шаблона
REF_ISN_OBJ_TEMPLATE	Целое число	REF ISN шаблона
ISN_USER	Целое число	Пользователь
CATEGORY	Строка (64)	Категория
NAME	Строка (64)	Название
WEIGHT	Целое число	Вес
FORMAT	Строка (64)	Формат
BODY	Большой двоичный объект	Содержимое

ORG_TYPE_CL

Справочник Типов организаций:

Таблица Г. 160

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN_LCLASSIF
WEIGHT	Целое число	Вес элемента
CLASSIF_NAME	Строка (64)	Наименование
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак лог удаления
NOTE	Строка (255)	Комментарий
INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Когда добавил
UPD_WHO	Целое число	Кто изменил
UPD_DATE	Дата и время	Когда изменил

ORGANIZ_CL

Справочник организаций:

Таблица Г. 161

Имя	Тип	Комментарии
DUE	Строка (248)	Код Дьюи организации
ISN_NODE	Целое число	ISN организации
ISN_HIGH_NODE	Целое число	Номер вышестоящ вершины
LAYER	Целое число	Номер уровня
IS_NODE	Целое число	Признак вершины
WEIGHT	Целое число	Вес элемента
MAXDUE	Строка (248)	MAX значение кода Дьюи
CLASSIF_NAME	Строка (255)	Наименование организации
CLASSIF_NAME_SEARCH	Строка (255)	Поиск наименования организации
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак лог удаления
FULLNAME	Строка (255)	Полное наименование
ZIPCODE	Строка (12)	Почтовый индекс
CITY	Строка (255)	Город
ADDRESS	Строка (255)	Почтовый адрес
MAIL_FOR_ALL	Целое число	Признак использования E_MAIL для всех представителей
NOTE	Строка (255)	Комментарий
NEW_RECORD	Целое число	Признак новой записи
OKPO	Строка (16)	ОКПО
INN	Строка (64)	ИНН
ISN_REGION	Целое число	Регион
OKONH	Строка (16)	ОКОНХ
LAW_ADRESS	Строка (255)	Юридический Адресс

Имя	Тип	Комментарии
ISN_ORGANIZ_TYPE	Целое число	Форма Собственности
SERTIFICAT	Строка (255)	Регистрационное свидетельство
ISN_ADDR_CATEGORY	Целое число	Категория адресата
CODE	Строка (4)	поле для формирования выписок для ЦБ
OGRN	Строка (64)	ОГРН
TERM_EXEC	Целое число	Срок исполнения РК
TERM_EXEC_TYPE	Целое число	Срок исполнения РК в каких днях
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил

PACK_ITEMS

Содержимое пакетов реестров вн отправки:

Таблица Г. 162

Имя	Тип	Комментарии
ISN_PACKITEM	Целое число	системный идентификатор пакета
ISN_PACKAGE	Целое число	системный идентификатор
ORDER_NUM	Целое число	порядковый номер
ISN_DOC	Целое число	документ
KIND_DOC	Целое число	вид документа
ISN_REF_SEND	Целое число	ISN адресата
CONSISTS	Строка (255)	Состав документа
DOC_NAME	Строка (64)	Наименование документа

PASS_STOP_LIST

Стоп слова для паролей пользователей:

Таблица Г. 163

Имя	Тип	Комментарии
ISN_PASS_STOP_LIST	Целое число	ISN записи
CLASSIF_NAME	Строка (64)	Название

PATCH_HISTORY

История Патчей:

Таблица Г. 164

Имя	Тип	Комментарии
RELISE_NUM	Целое число	Номер патча
SCHEMA_NUM	Целое число	Номер схемы
RELISE_DATE	Дата и время	Дата выполнения
RELISE_DESC	Строка (255)	Описание
APPLY_DATE	Дата и время	Дата применения обновления

PATCH_HISTORY_CUSTOM

История патчей для спецскриптов:

Таблица Г. 165

Имя	Тип	Комментарии
RELISE_NUM	Целое число	Номер версии
RELISE_DATE	Дата и время	Дата сборки
RELISE_DESC	Строка (255)	Описание
APPLY_DATE	Дата и время	Дата применения обновления

PLUGINS

Плагины:

Таблица Г. 166

Имя	Тип	Комментарии
WINDOW	Строка (24)	окно плагина
MENU_PATH	Строка (24)	место в меню окна куда входит пункт запускающий плагин
OBJ	Строка (24)	название класса плагина
LIBRARY	Строка (24)	название библиотеки в которой класс содержится
MENU_TEXT	Строка (255)	текст пункта меню

MENU_TAG	Строка (255)	свойство tag пункта меню
MENU_TBI_VISIBLE	Целое число	нужна ли кнопка на тулбаре окна запускаящая плагин
MENU_TBI_NAME	Строка (255)	картинка на кнопке на тулбаре
MENU_TBI_TEXT	Строка (255)	всплывающая подсказка на тулбаре
MENU_TBI_ORDER	Целое число	позиция кнопки на тулбаре
MENU_TBI_BARINDEX	Целое число	на каком тулбаре делать кн
INIT_STRING	Строка (2000)	строка инициализации плагина

PRINT_FILE_FORMAT

Формат печати:

Таблица Г. 167

Имя	Тип	Комментарии
PRINT_FILE_FORMAT_ID	Целое число	Идентификат
PRINT_FORM_ID	Целое число	Идентификатор
FORMAT	Целое число	Формат файла печати
WEIGHT	Целое число	Вес
IS_DEFAULT	Целое число	Формат по умолчанию

PRINT_FORM

Форма печати:

Таблица Г. 168

Имя	Тип	Комментарии
ID	Целое число	Идентификатор
NAME	Строка (64)	Название
UI_NAME	Строка (64)	Имя на экране
DESCRIPTION	Строка (255)	Описание
KIND	Целое число	Вид
PRINT_FORM_CATEGORY_ID	Целое число	Категория
IS_DELETED	Целое число	Признак лог удаления
IS_PROTECTED	Целое число	Запрет удаления

IS_DEFAULT	Целое число	Используется по умолчанию
WEIGHT	Целое число	Вес
TEMPLATE_FILE	VARBINARY (MAX)	Файл шаблона
TEMPLATE_FILE_EXTENSION	Строка (64)	Расширение
TEMPLATE_FILE_TIMESTAMP	Дата и время	Время изменения файла шаблона

PRINT_FORM_CATEGORY

Категории печатных форм:

Таблица Г. 169

Имя	Тип	Комментарии
PRINT_FORM_CATEGORY_ID	Целое число	Идентификатор записи
NAME	Строка (64)	Наименование
WEIGHT	Целое число	Вес
NOTE	Строка (255)	Примечание
DATA_TYPE	Строка (64)	Тип данных
LIST_FLAG	Целое число	Флаг списка

PRJ_CURR_PRJ

Текущая версия РКПД:

Таблица Г. 170

Имя	Тип	Комментарии
ISN_PRJ_BATCH	Целое число	ISN пакета версий проекта документов
ISN_PRJ_CURR	Целое число	ISN текущей версии проекта документов

PRJ_DEFAULT

Список идентификаторов реквизитов РКПД и их описание:

Таблица Г. 171

Имя	Тип	Комментарии
DEFAULT_ID	Строка (255)	Идентификатор реквизита
DEFAULT_TYPE	Строка (1)	Тип реквизита
DESCRIPTION	Строка (255)	Описание реквизита

CLASSIF_ID	Целое число	ссылки реквизита
------------	-------------	------------------

PRJ_DEFAULT_VALUE

Правила заполнения реквизитов РКПД:

Таблица Г. 172

Имя	Тип	Комментарии
ISN_DOCGROUP	Целое число	Код группы документов
DEFAULT_ID	Строка (255)	Идентификатор реквизита
VALUE	Строка (2000)	Значение реквизита

PRJ_ENDORSE_TASK

Задача на визирование ПД:

Таблица Г. 173

Имя	Тип	Комментарии
ISN_PROCESS_TASK	Целое число	ссылка на Задачу бизнес процесса
ISN_PRJ_VISA_SIGN	Целое число	Ссылка на Визу проставленную по данной задаче
AUTO_COMPLETE_ENDORSE	Целое число	Автоматическое завершение визирования
ALLOW_FILE_CHANGE	Целое число	Разрешить изменение файлов

PRJ_EXEC

Исполнители проекта документа:

Таблица Г. 174

Имя	Тип	Комментарии
ISN_PRJ_EXEC	Целое число	ISN исполнителя проекта
ISN_PRJ	Целое число	ISN проекта
DUE_PERSON	Строка (248)	DUE должностного лица
ORDERNUM	Целое число	№ в списке
ADDINFO	Строка (255)	Дополнительная информация исполнителя(роли)
CAN_MANAGE_EXEC	Целое число	Флаг права Управление Исполнителями
CAN_WORK_WITH_PRJ	Целое число	Флаг права Работа с РКПД

Имя	Тип	Комментарии
CAN_WORK_WITH_FILES	Целое число	Флаг права Работа с файлами РКПД
CAN_MANAGE_APPROVAL	Целое число	Флаг права Организация согласования и утверждения
IS_AUTHOR	Целое число	Признак "Автор проекта"
IS_EXEC	Целое число	Признак "Является исполнителем проекта"
CAN_MANAGE_PROCESS	Целое число	Признак "управление процессом"
CAN_MANAGE_ENDORSE	Целое число	Признак "Управление визирующими ПД"
CAN_MANAGE_SIGN	Целое число	Признак "Управление подписывающими ПД"
FOLDER_FLAG	Целое число	Должна ли быть запись о проекте в папке кабинета
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил

PRJ_FOLDER_ITEM

РКПД в папках кабинета:

Таблица Г. 175

Имя	Тип	Комментарии
ISN_FOLDER_ITEM	Целое число	ISN элемента папки
ISN_PRJ	Целое число	ISN проекта
ISN_PRJ_EXEC	Целое число	ISN исполнителя проекта
ISN_PRJ_VISA_SIGN	Целое число	ISN визы или подписи проекта
ISN_FOLDER	Целое число	ISN папки
GREEN_FLAG	Целое число	"Нетронутый элемент папки"
INS_WHO	Целое число	ISN пользователя поместившего элемент в папку
INS_DATE	Дата и время	Когда добавили

BOSS_VIEW	Целое число	Запись видна руководителю
ISN_PROCESS_TASK_COMPETITOR	Целое число	ISN конкурентного исполнителя

PRJ_FORUM

Форум РКПД:

Таблица Г. 176

Имя	Тип	Комментарии
ISN_PRJ_FORUM	Целое число	ISN форума РКПД
ISN_PRJ	Целое число	ISN РКПД
THEME_TEXT	Строка (255)	Наименование темы для обсуждения
FORUM_TEXT	Строка (2000)	Текст мнения
INS_DATE	Дата и время	Дата и время добавления мнения
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время редактирования мнения

PRJ_NUMCREATION

Счетчики номерообразования РКПД:

Таблица Г. 177

Имя	Тип	Комментарии
ISN_DOCGROUP	Целое число	Номер группы
YEAR_NUMBER	Целое число	Год
CURRENT_NUMBER	Целое число	Текущий номер
ISN_NUM_BASE	Целое число	ISN объекта
FLAG_MAX	Целое число	Признак макс номера
USER_ID	Строка (255)	Ид пользователя
LOCK_TIME	Дата и время	Время захвата записи

PRJ_PREP_TASK

Задача на подготовку ПД:

Таблица Г. 178

Имя	Тип	Комментарии
-----	-----	-------------

Имя	Тип	Комментарии
ISN_PROCESS_TASK	Целое число	Ссылка на Задачу бизнес процесса
CAN_READ_PRJ_RC	Целое число	Тип изменения права Чтение ПД
CAN_WORK_WITH_PRJ	Целое число	Тип изменения права Работа с ПД
CAN_WORK_WITH_FILES	Целое число	Тип изменения права Работа с файлами ПД
CAN_MANAGE_EXEC	Целое число	Тип изменения права Управление исполнителями ПД
CAN_MANAGE_ENDORSE	Целое число	Тип изменения права Управление визирующими ПД
CAN_MANAGE_SIGN	Целое число	Тип изменения права Управление подписывающими ПД
CAN_MANAGE_PROCESS	Целое число	Тип изменения права Управление процессом
AUTO_COMPLETE_PREPARATION	Целое число	AUTO_COMPLETE_PREАвтоматическое завершение подготовкиPARATION
ISN_PRJ_EXEC	Целое число	ISN исполнителя

PRJ_RC

Проекты документов (РКПД):

Таблица Г. 179

Имя	Тип	Комментарии
ISN_PRJ	Целое число	ISN проекта
ISN_PRJ_BATCH	Целое число	ISN пакета версий
PRJ_VERSION	Целое число	Номер версии
CURRENT_FLAG	Целое число	"Текущая версия"
PRJ_STAGE	Целое число	Код этапа работы над проектом
DUE_DOCGROUP	Строка (248)	DUE группы документов
SECURLEVEL	Целое число	Гриф доступа

Имя	Тип	Комментарии
FREE_NUM	Строка (64)	Номер проекта
FREE_NUM_SEARCH	Строка (64)	Номер проекта для поиска
PRJ_DATE	Дата и время	Дата проекта
PRJ_YEAR	Целое число	Год проекта
ORDER_NUM	Целое число	№ в группе
PLAN_DATE	Дата и время	Плановая дата завершения работы над проектом
ANNOTAT	Строка (2000)	Аннотация
CONSISTS	Строка (255)	Состав проекта
SPECIMEN	Строка (64)	Экземпляльность
NOTE	Строка (2000)	Примечание
AUTO_REG	Целое число	Авторегистрация проекта
ISN_NOMENKL	Целое число	ISN номенклатуры дел
APPLY_EDS	Целое число	Применять ЭП для подписания проекта
APPLY2_EDS	Целое число	Применять ЭП для визирования проекта
APPLY_EXEC_EDS	Целое число	Применять ЭП для исполнителя проекта
DEL_AFTER_REG	Целое число	Удалять РК проекта после регистрации
FILE_COUNT	Целое число	Количество файлов
FILE_COUNT_VISIBLE	Целое число	Количество видимых файлов
E_DOCUMENT	Целое число	Признак электронного документа
FREE_NUM_SORT	Строка (255)	Номер для сортировки
ISN_PRJ_STAGE	Целое число	Ссылка на статус РКПД
ISN_DOCVID	Целое число	Вид документа
BUSINESS_VERSION	Целое число	Бизнес версия
INS_DATE	Дата и время	Дата и время создания

Имя	Тип	Комментарии
INS_WHO	Целое число	Кто создал

PRJ_REF_RUBRIC

Рубрики РКПД:

Таблица Г. 180

Имя	Тип	Комментарии
ISN_PRJ_REF_RUBRIC	Целое число	ISN темы
ISN_REF_PRJ	Целое число	ISN РКПД
DUE_RUBRIC	Строка (248)	Код Дьюи темы
ORDERNUM	Целое число	№ в списке
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил

PRJ_REF_SEND

Адресаты проекта документа:

Таблица Г. 181

Имя	Тип	Комментарии
ISN_PRJ_REF_SEND	Целое число	ISN адресата
KIND_DOC	Целое число	Вид документа
ISN_REF_DOC	Целое число	ISN проекта
ISN_CITIZEN	Целое число	ISN адресата гражданина
DUE_ORGANIZ	Строка (248)	DUE адресата организации
DUE_DEP	Строка (248)	DUE внутреннего адресата
ISN_CONTACT	Целое число	ISN контакта
SEND_PERSON	Строка (64)	Кому адресован
SENDING_TYPE	Целое число	Тип отправки
ORDERNUM	Целое число	№ в списке
CONSISTS	Строка (255)	Состав документа

NOTE	Строка (255)	Примечание
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил
ISN_DELIVERY	Целое число	Вид отправки

PRJ_RETURN

Возврат РКПД исполнителю:

Таблица Г. 182

Имя	Тип	Комментарии
ISN_PRJ_RETURN	Целое число	ISN записи возврата РКПД исполнителю
ISN_PRJ	Целое число	ISN РКПД
REASON	Строка (2000)	Причина возврата
INS_WHO	Целое число	Кто создал
INS_DATE	Дата и время	Дата и время создания
RETURN_DATE	Дата и время	Дата возврата

PRJ_SIGN_TASK

Задача на подпись ПД:

Таблица Г. 183

Имя	Тип	Комментарии
ISN_PROCESS_TASK	Целое число	Ссылка на Задачу бизнес процесса
ISN_PRJ_VISA_SIGN	Целое число	Ссылка на Подпись проставленную по данной задаче
ALLOW_FILE_CHANGE	Целое число	Разрешить изменение файлов

PRJ_STAGE_CL

Справочник Статус проекта документа:

Таблица Г. 184

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	Идентификатор

CLASSIF_NAME	Строка (64)	Название
NOTE	Строка (2000)	Описание
WEIGHT	Целое число	Порядок
PRJ_STAGE	Целое число	для совместимости
UPD_DATE	Строка (64)	Дата изменения
UPD_WHO	Целое число	Кто изменил

PRJ_VISA_SIGN

Подписи и визы проекта:

Таблица Г. 185

Имя	Тип	Комментарии
ISN_PRJ_VISA_SIGN	Целое число	ISN подписи или визы проекта
ISN_PRJ	Целое число	ISN проекта
KIND	Целое число	Вид записи
DUE_PERSON	Строка (248)	DUE визирующего или подписывающего
DUE_PERSON_PARENT	Строка (248)	DUE запросившего визу или подпись
WAIT_FLAG	Целое число	Номер посылки на визирование или подпись
WEIGHT_SND	Целое число	Вес в посылке
TERM	Целое число	Срок ответа
TERM_FLAG	Целое число	Ед измерения срока
RECEIVE_DATE	Дата и время	Дата получения на визирование или подпись
TERM_DATE	Дата и время	Плановая дата и время окончания срока ответа
REP_DATE	Дата и время	Дата ответа
SIGN_TYPE	Целое число	Код типа подписи
ISN_VISA_TYPE	Целое число	ISN типа визы
REP_TEXT	Строка (2000)	Текст ответа
RESERVE_FOLDER_FLAG	Целое число	"Оставить в папке"

Имя	Тип	Комментарии
NOTE	Строка (2000)	Примечание
WEIGHT	Целое число	Вес в списке
ALLOW_FILE_CHANGE	Целое число	Разрешено изменение файлов
PROT_COMMENT	Строка (255)	Комментарий к протоколу
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил

PROCESS_TASK

Задача бизнес-процесса:

Таблица Г. 186

Имя	Тип	Комментарии
ISN_PROCESS_TASK	Целое число	Идентификатор
ISN_OBJ	Целое число	Ссылка на связанный с задачей объект
KIND_OBJ	Целое число	Тип связанного с задачей объекта
ISN_INSTANCE_COMP_EXEC	Целое число	Ссылка на Компонент бизнес процесса
ISN_TASK_STATUS	Целое число	Ссылка на справочник Статусы задачи
IS_COMPETITIVE	Целое число	Признак Конкурентной задачи
ISN_EXEC	Целое число	Исполнитель
PLAN_DATE	Дата и время	Плановая дата
NOTE	Строка (2000)	Описание
APPLY_EDS	Целое число	Применять ЭлПодпись
ISN_PARENT_TASK	Целое число	Исходная задача
ISN_INSTANCE_COMPONENT	Целое число	Экземпляр компонента
INS_DATE	Дата и время	Дата создания

INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата изменения
UPD_WHO	Целое число	Кто изменил

PROCESS_TASK_COMPETITOR

Конкурентные исполнители задачи бизнес-процесса:

Таблица Г. 187

Имя	Тип	Комментарии
ISN_PROCESS_TASK_COMPETITOR	Целое число	ISN записи
ISN_PROCESS_TASK	Целое число	Ссылка на Задачу бизнес процесса
ISN_EXEC	Целое число	Исполнитель
ISN_INSTANCE_COMP_EXEC	Целое число	ISN экз комп

PROCESS_TASK_STATUS_CL

Справочник Статус задачи:

Таблица Г. 188

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	Идентификатор
CLASSIF_NAME	Строка (64)	Название
NOTE	Строка (2000)	Описание
WEIGHT	Целое число	Порядок
IS_FINAL	Целое число	Финальный
UPD_DATE	Дата и время	Дата изменения
UPD_WHO	Целое число	Кто изменил

PROGRAMS

Таблица Г. 189

Имя	Тип	Комментарии
VERSION	NUMBER(10)	Номер версии
APP_PART	VARCHAR2(24)	Приложение
CREATE_DATE	DATE	Дата выпуска

DESCRIPTION	VARCHAR2(255)	Описание
CLEAR	NUMBER(1)	Обновлять с очисткой каталога

PROT

Протокол работы пользователей:

Таблица Г. 190

Имя	Тип	Комментарии
TABLE_ID	Строка (3)	Идентификатор модифицируемой таблицы
OPER_ID	Строка (1)	Идентификатор операции
SUBOPER_ID	Строка (1)	Идентификатор операции в дочерней таблице
OPER_DESCRIBE	Строка (3)	Идентификатор дочерней таблицы
REF_ISN	Целое число	ISN записи модифицируемой таблицы
TIME_STAMP	Дата и время	Время модификации
USER_ISN	Целое число	ISN пользователя
OPER_COMMENT	Строка (255)	Комментарий к операции
ISN_PROT_INFO	Целое число	ISN доп информации

PROT_FILE_SSCAN

Файлы протокола:

Таблица Г. 191

Имя	Тип	Комментарии
ISN_PROT	Целое число	ISN протокола
NUM_PP	Целое число	Номер пп файла в протоколе
SCAN_NUM	Целое число	Номер сканера
NAME_FILE	Строка (24)	Имя файла
STATUS_FILE	Целое число	Статус файла
CNT_LIST_FILE	Целое число	Кол страниц в файле
DATE_DEL	Дата и время	Дата и время последней обработки файла
ISN_DEL	Целое число	ISN USER выполневшего

		последнюю обработку
--	--	---------------------

PROT_INFO

Дополнительная информация протокола:

Таблица Г. 192

Имя	Тип	Комментарии
ISN_PROT_INFO	Целое число	ISN записи
INFO	Текст	Информация
ISN_CHANNEL	Целое число	ISN_канала

PROT_NAME

Перечень протоколируемых операций:

Таблица Г. 193

Имя	Тип	Комментарии
TABLE_ID	Строка (3)	Идентификатор таблицы
OPER_ID	Строка (1)	Идентификатор операции
SUBOPER_ID	Строка (1)	идентификатор подоперации
OPER_DESCRIBE	Строка (3)	Описатель операции
DESCRIPTION	Строка (64)	Описание
NOTE	Строка (255)	Примечание
VIEW_PARM_POSITION	Целое число	Позиция в параметре пользователя

PROT_STREAM_SCAN

Протокол поточного сканирования:

Таблица Г. 194

Имя	Тип	Комментарии
ISN_PROT	Целое число	ISN протокола
ISN_RC	Целое число	ISN Рк
KIND_RC	Целое число	Вид Рк
DATE_PRINT	Дата и время	Дата и время печати штрихкода
ISN_USER_BARCODE	Целое число	ISN USER напечатавшего штрихкод
DATE_SCAN	Дата и время	Дата и время сканирования

Имя	Тип	Комментарии
ISN_USER_SCAN	Целое число	ISN USER отсканировавшего документ
CNT_LIST	Целое число	Кол отсканированных страниц
CNT_SC_T	Целое число	Кол текст файлов
CNT_SC_G	Целое число	Кол граф файлов
CNT_READY_T	Целое число	Кол текст файлов готовых к записи
CNT_READY_G	Целое число	Кол граф файлов готовых к записи
CNT_DEL_T	Целое число	Кол удаленных текст файлов
CNT_DEL_G	Целое число	Кол удаленных граф файлов
CNT_ATT_T	Целое число	Кол прикрепленных текст файлов
CNT_ATT_G	Целое число	Кол прикрепленных граф файлов
CNT_NATT_T	Целое число	Кол неприкрепившихся текст файлов
CNT_NATT_G	Целое число	Кол неприкрепившихся граф файлов
CNT_SAVE_NATT_T	Целое число	Кол сохраненных неприкр текст файлов
CNT_SAVE_NATT_G	Целое число	Кол сохраненных неприкр граф файлов
CLOSE_PROT	Целое число	Признак закрытия протокольной записи
LOCK_FLAG	Целое число	Признак защиты от редактирования
NOTE	Строка (64)	Примечание
DATE_WRITE_DB	Дата и время	Дата и время первой записи в БД
EHD_DOCTYPE	Строка (32)	Тип документа ЭХД

RAC_ADJUST

Отслеживание пересчета РК в картотеках:

Таблица Г. 195

Имя	Тип	Комментарии
DUE_CARD_ADJUST	Строка (248)	Картотека пересчета
DUE_CARD	Строка (248)	Измененная картотека
WATERMARK	Целое число	Последняя измененная РК
TIME_STAMP	Дата и время	Время заявки
OPER	Строка (24)	Операция

READ_PROT

Журнал отметок о прочтении:

Таблица Г. 196

Имя	Тип	Комментарии
USER_ISN	Целое число	Системный номер пользователя
TABLE_ID	Строка (1)	Идентификатор таблицы
REF_ISN	Целое число	Системный номер объекта
TIME_STAMP	Дата и время	Дата и время чтения пользователем объекта

REESTR_NEW

Реестры внешней отправки:

Таблица Г. 197

Имя	Тип	Комментарии
ISN_REESTR	Целое число	Системный идентификатор
ISN_REESTR_TYPE	Целое число	Тип реестра
FREE_NUM	Строка (64)	Номер реестра
ISN_DELIVERY	Целое число	Вид отправки
SEND_DATE	Дата и время	Дата отправки
REESTR_YEAR	Целое число	год отправки
STATUS	Целое число	статус
TAKE_CODES	Целое число	Присвоены по коды
RTM	Строка (255)	RTM
ISN_USER	Целое число	Ответственный пользователь

DUE_EXPEDITION	Строка (248)	Подразделение отправки
NOTE	Строка (255)	Комментарий
IS_PERSONAL	Целое число	Категория реестра
NUMCREATION_FLAG	Целое число	Флаг номерообразования
PACKAGE_CNT	Целое число	Количество пакетов
DOC_CNT	Целое число	Количество документов
INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Дата и время создания

REESTRTYPE_CL

Справочник Типов реестров:

Таблица Г. 198

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN типа реестра
WEIGHT	Целое число	Вес элемента
CLASSIF_NAME	Строка (64)	Имя типа реестра
PROTECTED	Целое число	Пр запрета удаления
DELETED	Целое число	Пр лог удаления
ISN_ADDR_CATEGORY	Целое число	Категория адресата
ISN_DELIVERY	Целое число	ISN вид доставки
FLAG_TYPE	Целое число	Признак номерообразования
EMERGENCY	Строка (64)	Срочность
IMPOTANCE	Строка (64)	Важность
GROUP_MAIL	Целое число	Партионная почта
NOTE	Строка (255)	Комментарий
INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Когда добавил
UPD_WHO	Целое число	Кто изменил
UPD_DATE	Дата и время	Когда изменил

REF_ACCESS_CARD

Принадлежность РК к картотекам:

Таблица Г. 199

Имя	Тип	Комментарии
ISN_REF_DOC	Целое число	ISN документа
DUE	Строка (248)	Код Дьюи ДЛ (картотеки)
KIND_DOC	Целое число	Вид документа
DATE_CARD	Дата и время	Дата входа в картотеку
CONTROL_STATE	Целое число	На контроле
ISN_RESOLUTION	Целое число	ISN резолюции(первой) (не используется)
HOLD	Целое число	Не удалять картотеку при пересчете

REF_CONTEXT

Контекст объекта для поиска:

Таблица Г. 200

Имя	Тип	Комментарии
ISN_REF_CONTEXT	Целое число	ISN контекста
ISN_REF_DOC	Целое число	ISN объекта
KIND_DOC	Целое число	Вид объекта
FILE_EXTENSION	Строка (255)	Расширение файла
REG_DATE	Дата и время	Дата регистрации
CONTEXT	Большой двоичный объект	Контекст
ORDER_NUM	Целое число	Порядковый номер
FREE_NUM_SEARCH	Строка (64)	Текстовый номер документа

REF_CORRESP

Корреспонденты и Сопроводительные документов РК:

Таблица Г. 201

Имя	Тип	Комментарии
ISN_REF_CORRESP	Целое число	ISN корреспондента
ISN_DOC_INP	Целое число	ISN документа

Имя	Тип	Комментарии
DUE_ORGANIZ	Строка (248)	Код Дьюи организации
ISN_CONTACT	Целое число	ISN контакта
KIND_DOC	Целое число	Вид РК (дубль)
CORRESP_KIND	Целое число	Вид записи
ORDERNUM	Целое число	№ в списке
CORRESP_DATE	Дата и время	Исходящая дата
CORRESP_NUM	Строка (64)	Исходящий номер
CORRESP_SIGN	Строка (255)	Кто подписал
TEL_NUM	Строка (64)	№ телегр_мы письма
ANNOTAT	Строка (2000)	Краткое содержание
CORRESP_NUM_SEARCH	Строка (64)	Не используется
CORRESP_SIGN_SEARCH	Строка (255)	Кто подписал верх рег
DOC_NAME	Строка (64)	Название документа
CONSISTS	Строка (64)	Состав
ANNOTAT_1	Строка (255)	Аннотация_1
NOTE	Строка (2000)	Примечание
NOTE_1	Строка (255)	Примечание_1
SENDER_FLAG	Целое число	признак отправителя электронного сообщения
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил

REF_FILE

Файлы РК и РКПД:

Таблица Г. 202

Имя	Тип	Комментарии
ISN_REF_FILE	Целое число	ISN файла

Имя	Тип	Комментарии
ISN_FILE_TYPE	Целое число	ISN типа файла
ISN_REF_DOC	Целое число	ISN документа
ORDERNUM	Целое число	№ в списке
NAME	Строка (255)	Название файла
STORAGE	Целое число	Тип хранения
PATH	Строка (255)	Путь
FILESIZE	Целое число	Размер
FREE_NUM	Строка (24)	Номер документа
DOC_DATE	Дата и время	Дата документа
DESCRIPTION	Строка (255)	Комментарий
CATEGORY	Строка (248)	Категория
KIND_DOC	Целое число	Вид документа
SECURLEVEL	Целое число	Гриф доступа
ACCESS_MODE	Целое число	Вид доступа
LOCK_FLAG	Целое число	Флаг защиты
ISN_USER_LOCK	Целое число	Кто защитил
ISN_USER_EDIT	Целое число	Кто редактировал
SCAN_NUM	Целое число	SCAN_NUM
EDS_CNT	Целое число	Количество подписей
GR_STORAGE	Целое число	Группа хранения
DONTDEL_FLAG	Целое число	Запрещено удалять
ISN_USER_DONTDEL	Целое число	Кто последний запретил удалять
IS_HIDDEN	Целое число	Признак скрытого файла
APPLY_EDS	Целое число	Применять ЭП
SEND_ENABLED	Целое число	Рассылать
UPD_CONTENT_DATE	Дата и время	Дата и время обновления

Имя	Тип	Комментарии
		содержимого
EPGU_FLAG	Целое число	Флаг ЕПГУ
ISN_FILELINK	Целое число	ISN ссылки на файл
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
TAG	Строка (255)	Тег
ISN_REF_FILE_BATCH	Целое число	ISN пакета файла
LOCK_DATE	Дата и время	Дата блокировки
REF_FILE_TYPE	Целое число	Тип файла
VERSION_NUM	Целое число	Номер версии
ISN_OBJECT	Целое число	ISN объекта
VERSION_NOTE	Строка (255)	Описание версии
ISN_FILE_CATEGORY	Целое число	Категория файла
IS_CURRENT	Целое число	Признак текущего файла
ISN_REQUEST	Целое число	Ссылка на SSCAN_REQUEST
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил

REF_FILE_ACCESS

Доступ к файлам для ДЛ:

Таблица Г. 203

Имя	Тип	Комментарии
ISN_REF_FILE_ACCESS	Целое число	ISN записи доступа
ISN_REF_FILE	Целое число	ISN файла
DUE_DEP	Строка (248)	Должностное лицо

REF_FILE_EDS

ЭП файлов:

Таблица Г. 204

Имя	Тип	Комментарии
-----	-----	-------------

ISN_REF_FILE_EDS	Целое число	ISN ЭП файла
ISN_REF_FILE	Целое число	ISN файла
ID_CERTIFICATE	Строка (500)	Идентификатор сертификата
CERTIFICATE_OWNER	Строка (2000)	Владелец сертификата
SIGNING_DATE	Дата и время	Дата подписи
ISN_USER	Целое число	Пользователь сделавший запись
ISN_SIGN_KIND	Целое число	ISN вида подписи
DUE_PERSON	Строка (248)	DUE подписывающего
SIGN_TEXT	Строка (255)	Текст подписи
EDS_DATA	Большой двоичный объект	Содержимое подписи

REF_LETTER

Заявитель письма:

Таблица Г. 205

Имя	Тип	Комментарии
ISN_REF_LETTER	Целое число	ISN заявителя
ISN_CITIZEN	Целое число	ISN гражданина
ISN_DOC_INP	Целое число	ISN документа
NEEDANSWER	Целое число	Требование ответа
ORDERNUM	Целое число	№ в списке
TEL_NUM	Строка (64)	Номер телеграммы письма
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил

REF_LINK

Связки документов:

Таблица Г. 206

Имя	Тип	Комментарии
-----	-----	-------------

Имя	Тип	Комментарии
ISN_REF_LINK	Целое число	ISN связи
ISN_CLLINK	Целое число	ISN типа связи
KIND_DOC	Целое число	Вид документа
ISN_REF_DOC	Целое число	ISN документа
KIND_LINKED_DOC	Целое число	Вид св документа
ISN_LINKED_DOC	Целое число	ISN св документа
LINKED_NUM	Строка (255)	Висячая связка
URL_STR	Строка (255)	URL
ORDERNUM	Целое число	№ в списке
REG_COUNT	Целое число	Использована при номераобразовании
TRANSPARENT	Целое число	Есть доступ к связанному объекту
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил

REF_RUBRIC

Рубрики документов:

Таблица Г. 207

Имя	Тип	Комментарии
ISN_REF_RUBRIC	Целое число	ISN темы
KIND_DOC	Целое число	Вид документа
ISN_REF_DOC	Целое число	isn документа
DUE_RUBRIC	Строка (248)	Код Дьюи темы
ORDERNUM	Целое число	№ в списке
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал

UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил

REF_SEND

Адресаты РК:

Таблица Г. 208

Имя	Тип	Комментарии
ISN_REF_SEND	Целое число	ISN адресата
KIND_DOC	Целое число	Вид документа
ISN_REF_DOC	Целое число	ISN документа
ISN_CITIZEN	Целое число	ISN гражданина
DUE_DEP	Строка (248)	Внутренний адресат
ISN_REESTR	Целое число	ISN реестра
SEND_PERSON	Строка (64)	Кому адресован
ORDERNUM	Целое число	№ в списке
ISN_DELIVERY	Целое число	ISN вид доставки
SEND_DATE	Дата и время	Дата отправки
ISN_SENDUSER	Целое число	Пользователь отправивший
CONSISTS	Строка (255)	Состав документа
ISN_CONTACT	Целое число	ISN контакта
DUE_ORGANIZ	Строка (248)	Код Дьюи организации
FLAG_ORIGINAL	Целое число	Оригинал копия
NOTE	Строка (2000)	Примечание
SENDING_TYPE	Целое число	Тип отправки
REG_N	Строка (64)	Рег номер адресата
REG_DATE	Дата и время	Дата регистрации адресата
ANSWER	Целое число	Требуется ли квитанция о регистрации
ANSWER_DATE	Дата и время	Дата квитанции
ISN_ADDRESS	Целое число	ISN адреса

Имя	Тип	Комментарии
DUE_EXPEDITION	Строка (248)	Подразделение отправки
BLANK_NUM	Строка (64)	Номер бланка
DUE_SENDING_DEP	Строка (248)	Где создан
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил
ISN_MESSAGE	Целое число	ISN сообщения

REF_SOISP

Соисполнители документов:

Таблица Г. 209

Имя	Тип	Комментарии
ISN_REF_SOISP	Целое число	ISN соисполнителя
ISN_DOC_OUT	Целое число	ISN документа
DUE_ORGANIZ	Строка (248)	Код Дьюи организации
ISN_CONTACT	Целое число	ISN контакта
SOISP_NUM	Строка (64)	Исходящий номер
SOISP_DATE	Дата и время	Исходящая дата
SOISP_PERSON	Строка (255)	Кто подписал
ORDERNUM	Целое число	№ в списке
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил

REF_UFOLDER

Элементы личных папок:

Таблица Г. 210

Имя	Тип	Комментарии
-----	-----	-------------

ISN_REF_UFOLDER	Целое число	ISN папки
ISN_REF_DOC	Целое число	ISN документа
KIND_DOC	Целое число	Вид документа
CREATE_DATE	Дата и время	Дата добавления
CREATE_WHO	Целое число	Кто добавил
NOTE	Строка (255)	Комментарий
UPD_DATE	Дата и время	Дата изменения
UPD_WHO	Целое число	Кто изменил

REF_VISA

Визы РК:

Таблица Г. 211

Имя	Тип	Комментарии
ISN_REF_VISA	Целое число	ISN визы
ISN_DOC_OUT	Целое число	ISN документа
DUE_PERSON	Строка (248)	Код Дьюи ДЛ
VISA_DATE	Дата и время	Дата визирования
ORDERNUM	Целое число	№ в списке
NOTE	Строка (2000)	Комментарий
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил

REGION_CL

Справочник Регионов:

Таблица Г. 212

Имя	Тип	Комментарии
DUE	Строка (248)	Дьюи Регионов
ISN_NODE	Целое число	ISN Регионов
ISN_HIGH_NODE	Целое число	Ссылка на родительскую запись

Имя	Тип	Комментарии
LAYER	Целое число	Номер уровня
IS_NODE	Целое число	Признак вершины
WEIGHT	Целое число	Вес элемента
MAXDUE	Строка (248)	MAX значение кода Дьюи
CLASSIF_NAME	Строка (64)	Наименование
PROTECTED	Целое число	Пр запрета удаления
DELETED	Целое число	Пр лог удаления
NOTE	Строка (255)	Примечание
CODE	Целое число	Код региона
COD_OKATO	Строка (11)	Код OKATO
INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Когда добавил
UPD_WHO	Целое число	Кто изменил
UPD_DATE	Дата и время	Когда изменил

REMINDER

Напоминания исполнителям поручений:

Таблица Г. 213

Имя	Тип	Комментарии
ISN_REMINDER	Целое число	ISN напоминания
ISN_REPLY	Целое число	ISN исполнителя
INS_DATE	Дата и время	Дата создания
INS_WHO	Целое число	Кто создал
REMINDER_TEXT	Строка (255)	Текст напоминания
READ_DATE	Дата и время	Дата прочтения
READ_WHO	Целое число	Кто прочитал

REP_NUMCREATION

Запрашиваемые номера:

Таблица Г. 214

Имя	Тип	Комментарии
LAST_DATE	Дата и время	Последняя дата
CUR_NUMBER	Целое число	Текущий номер

REP_NUMCREATION2

Запрашиваемые номера2:

Таблица Г. 215

Имя	Тип	Комментарии
LAST_DATE	Дата и время	Последняя дата
CUR_NUMBER	Целое число	Текущий номер

REPEAT_RES

Правила повторения резолюции:

Таблица Г. 216

Имя	Тип	Комментарии
ISN_REPEAT_RES	Целое число	ISN записи
ISN_ACTIVE_RESOLUTION	Целое число	ISN активной резолюции
INTERVAL_TYPE	Целое число	Тип интервала
INTERVAL_SUBTYPE	Целое число	Подтип интервала
BASE_DATE	Дата и время	Базовая дата
IS_WORKDAY	Целое число	Рабочий день
DAY_NUMBER	Целое число	Номер дня
WEEK_NUMBER	Целое число	Номер недели
MONTH_NUMBER	Целое число	Номер месяца
WEEK_DAYS	Строка (7)	Дни недели
EXPIRE_DATE	Дата и время	Дата окончания повторов
REPEAT_COUNT	Целое число	Количество повторов
REPEAT_DATE	Дата и время	Дата повторения
INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Когда добавил
UPD_WHO	Целое число	Кто обновил

UPD_DATE	Дата и время	Когда обновил
----------	--------------	---------------

REPLY

Исполнители поручений:

Таблица Г. 217

Имя	Тип	Комментарии
ISN_REPLY	Целое число	ISN ответа
DUE	Строка (248)	DUE ДЛ исполнителя
ISN_RESOLUTION	Целое число	ISN резолюции
MAIN_FLAG	Целое число	Ответ исполнитель
RECEPT_DATE	Дата и время	Дата приема док_та
REPLY_TEXT	Строка (2000)	Текст ответа
REPLY_TEXT_1	Строка (255)	Текст ответа_1
PLAN_DATE	Дата и время	План дата исполнения
PLAN_DATE2	Дата и время	Плановая дата для не выполненного отчета
REPLY_DATE	Дата и время	Факт дата ответа
DELTA_DATE	Целое число	Нарушение срока
ISN_STATUS_REPLY	Целое число	ISN состояния исполнения (И)
IS_CABINET	Целое число	Пр наличия кабинета
ORDER_EXEC	Целое число	№ исполнителя
NOTE	Строка (255)	Примечание
REMINDER_TOTAL	Целое число	Всего напоминаний
REMINDER_UNREAD	Целое число	Непрочтенных напоминаний
INS_DATE	Дата и время	Дата и время создания
INS_WHO	Целое число	Кто создал
UPD_DATE	Дата и время	Дата и время обновления
UPD_WHO	Целое число	Кто обновил

RESOLUTION

Резолюции и пункты:

Таблица Г. 218

Имя	Тип	Комментарии
ISN_RESOLUTION	Целое число	ISN резолюции
KIND_RESOLUTION	Целое число	Вид поручения
KIND_DOC	Целое число	Вид документа
ISN_REF_DOC	Целое число	ISN документа
ISN_PARENT_RESOLUTION	Целое число	ISN родительской резолюции
LAYER	Целое число	Уровень подчиненности
WEIGHT	Целое число	Вес резолюции
ITEM_NUMBER	Строка (64)	Номер пункта
DUE	Строка (248)	DUE ДЛ автора
RESOLUTION_TEXT	Строка (2000)	Текст резолюции
ISN_CATEGORY	Целое число	ISN категории
CONFIDENTIAL	Целое число	"Конфиденциальная резолюция"
RESOLUTION_DATE	Дата и время	Дата резолюции
SEND_DATE	Дата и время	Дата рассылки
NOTIFY_AUTHOR	Целое число	Посылать автору
PLAN_DATE	Дата и время	Плановая дата
PLAN_DATE2	Дата и время	Плановая дата для пустой фактической
INTERIM_DATE	Дата и время	Промежуточная дата
FACT_DATE	Дата и время	Фактическая дата
DELTA_DATE	Целое число	Нарушение срока
DUE_CONTROLLER	Строка (248)	DUE ДЛ контролера
ACCEPT_CONTROL	Целое число	"Принято на контроль"
CONTROL_STATE	Целое число	На контроле
ISN_STATUS_EXEC	Целое число	ISN состояния исполнения (К)
SUMMARY	Строка (2000)	Ход исполнения

Имя	Тип	Комментарии
CASCADE_CONTROL	Целое число	"Снимать с контроля каскадно"
CONTROL_DUTY_FLAG	Целое число	"Резолюция отвечает за контрольность РК"
RESUME	Строка (2000)	Основание для снятия с контроля
LEFT_REZOLUTION	Целое число	Признак рассылки резолюции
CYCLE	Строка (1)	Признак рассылки исполнителям в кабинет автора
NOTE	Строка (255)	Примечание
APPROVED	Целое число	Флаг утверждения резолюции
ISN_RESPRJ_PRIORITY	Целое число	ISN приоритета проекта резолюции
RESPRJ_STATUS	Целое число	Статус проекта резолюции
ISN_REPEAT_RES	Целое число	ISN правила повторения
INS_WHO	Целое число	Кто создал запись
INS_DATE	Дата и время	Когда создал
UPD_DATE	Дата и время	Когда обновил

RESOLUTION_CARD

Принадлежность поручений картотекам:

Таблица Г. 219

Имя	Тип	Комментарии
ISN_RESOLUTION	Целое число	ISN резолюции
DUE	Строка (248)	DUE картотеки

RESOLUTION_CATEGORY_CL

Справочник Категорий поручений:

Таблица Г. 220

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN категории
WEIGHT	Целое число	Вес элемента
CLASSIF_NAME	Строка (64)	Наименование

Имя	Тип	Комментарии
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак лог удаления
NOTE	Строка (255)	Комментарий
PLAN_COUNT_FLAG	Целое число	Использовать для расчета срока поручения
PLAN_DAY_COUNT	Целое число	Количество дней
PLAN_DAY_TYPE	Целое число	Расчет в рабочих днях или календарных днях
PLAN_PROTECTED	Целое число	Запретить редактирование плановой даты
INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Когда добавил
UPD_WHO	Целое число	Кто изменил
UPD_DATE	Дата и время	Когда изменил

RESOLUTION_PRJ_COPY

Копии проектов резолюций

Таблица Г. 221

Имя	Тип	Комментарии
ISN_RESOLUTION_PRJ_COPY	Целое число	ISN записи
ISN_RESOLUTION	Целое число	ISN резолюции
COPY_TEXT	Текст	Копия текста
USER_ISN	Целое число	ISN сохранившего пользователя
TIME_STAMP	Дата и время	Дата Время сохранения

RESPRJ_PRIORITY_CL

Приоритеты проекта поручения:

Таблица Г. 222

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN приоритета проекта поручения
WEIGHT	Целое число	Вес элемента

Имя	Тип	Комментарии
CLASSIF_NAME	Строка (64)	Наименование
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак лог удаления
NOTE	Строка (255)	Комментарий
INS_WHO	Целое число	Кто создал
INS_DATE	Дата и время	Дата и время создания
UPD_WHO	Целое число	Кто обновил
UPD_DATE	Дата и время	Дата и время обновления

RESPRJ_STATUS_CL

Статусы проекта поручения:

Таблица Г. 223

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN элемента
WEIGHT	Целое число	Вес элемента
CLASSIF_NAME	Строка (64)	Наименование
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак лог удаления
NOTE	Строка (255)	Комментарий

RUBRIC_CL

Справочник Рубрикатор:

Таблица Г. 224

Имя	Тип	Комментарии
DUE	Строка (248)	Код Дьюи темы
ISN_NODE	Целое число	ISN темы
ISN_HIGH_NODE	Целое число	№ верхней вершины
LAYER	Целое число	Номер уровня
IS_NODE	Целое число	Признак вершины
WEIGHT	Целое число	Вес элемента

Имя	Тип	Комментарии
MAXDUE	Строка (248)	MAX значение кода Дьюи
CLASSIF_NAME	Строка (2000)	Наименование темы
FULLNAME	Строка (2000)	Полное название
PROTECTED	Целое число	Пр запрета удаления
DELETED	Целое число	Пр лог удаления
RUBRIC_CODE	Строка (248)	Код рубрики
CODE	Строка (64)	Код рубрики_
NOTE	Строка (255)	Комментарий
INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Когда добавил
UPD_WHO	Целое число	Кто изменил
UPD_DATE	Дата и время	Когда изменил

SECURITY_CL

Справочник Грифы доступа:

Таблица Г. 225

Имя	Тип	Комментарии
SECURLEVEL	Целое число	Гриф доступа
GRIF_NAME	Строка (64)	Наименование грифа
DELETED	Целое число	Признак лог удаления
PROTECTED	Целое число	Признак запрета удаления
WEIGHT	Целое число	Вес элемента
EDS_FLAG	Целое число	Требуется ЭП
ENCRYPT_FLAG	Целое число	Требуется шифрование
SEC_INDEX	Строка (24)	Индекс грифа
CONFIDENTIONAL	Целое число	ДСП файлы
NOTE	Строка (255)	Примечание
INS_WHO	Целое число	Кто создал

INS_DATE	Дата и время	Когда создал
UPD_WHO	Целое число	Кто изменил
UPD_DATE	Дата и время	Когда изменил

SEND_PACKAGE

Пакеты реестров внешней отправки:

Таблица Г. 226

Имя	Тип	Комментарии
ISN_PACKAGE	Целое число	системный идентификатор
ISN_REESTR	Целое число	Реестр
ORDER_NUM	Целое число	Номер пакета
ISN_CITIZEN	Целое число	ISN гражданина
DUE_ORGANIZ	Строка (248)	Код Дьюи организации
SEND_PERSON	Строка (64)	Кому адресован
IMPOTANCE	Строка (64)	Важность
EMERGENCY	Строка (64)	Срочность
ISN_DOPADDR	Целое число	Дополнительный адресс
BAR_CODE	Строка (16)	Почтовый код
ISN_ADDRESS	Целое число	ISN адреса
NOTE	Строка (255)	Примечание

SEV_ASSOCIATION

Индекс СЭВ для элементов справочников:

Таблица Г. 227

Имя	Тип	Комментарии
OBJECT_ID	Строка (255)	Идентификатор в БД
OBJECT_NAME	Строка (64)	Сущность в БД
GLOBAL_ID	Строка (255)	Глобальный идентификатор
OWNER_ID	Строка (255)	Владелец
SENDER_ID	Строка (255)	Отправитель документа

SEV_CHANNEL

Канал передачи сообщений:

Таблица Г. 228

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN канала передачи сообщений
CHANNEL_TYPE	Строка (64)	Тип канала
PARAMS	Текст	Параметры доставки
WEIGHT	Целое число	Вес элемента
CLASSIF_NAME	Строка (64)	Наименование канала передачи
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак логического удаления
NOTE	Строка (255)	Примечание
INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Когда добавил
UPD_WHO	Целое число	Кто изменил
UPD_DATE	Дата и время	Когда изменил

SEV_COLLISION

Коллизии СЭВ:

Таблица Г. 229

Имя	Тип	Комментарии
COLLISION_CODE	Целое число	номер коллизии
REASON_NUM	Целое число	номер причины возникновения коллизии
COLLISION_NAME	Строка (255)	название коллизии
RESOLVE_TYPE	Целое число	Способ разрешения выбранный пользователем для данной коллизии
DEFAULT_RESOLVE_TYPE	Целое число	Значение по умолчанию рекомендованное для данной коллизии
ALLOWED_RESOLVE_TYPES	Строка (255)	через запятую допустимые способы для данной коллизии

UPD_WHO	Целое число	Кто изменил
UPD_DATE	Дата и время	Когда изменил

SEV_PARTICIPANT

Участник СЭВ:

Таблица Г. 230

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN участника МЭДО
DUE_ORGANIZ	Строка (248)	DUE организации
ISN_CHANNEL	Целое число	Канал передачи сообщений
WEIGHT	Целое число	Вес элемента
CLASSIF_NAME	Строка (255)	Именование участника МЭДО
ADDRESS	Строка (255)	Адрес
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак логического удаления
NOTE	Строка (255)	Примечание
CRYPT	Целое число	Признак шифрования сообщений
INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Когда добавил
UPD_WHO	Целое число	Кто изменил
UPD_DATE	Дата и время	Когда изменил

SEV_PARTICIPANT_RULE

Используемые правила:

Таблица Г. 231

Имя	Тип	Комментарии
ISN_PARTICIPANT	Целое число	Участник МЭДО
ISN_RULE	Целое число	Правило
ORDERNUM	Целое число	Порядок выбора правила

SEV_REPORT

Доклады:

Таблица Г. 232

Имя	Тип	Комментарии
ISN_REPORT	Целое число	ISN доклада
ISN_DOC	Целое число	Документ
KIND_DOC	Целое число	Вид документа
DUE_ORGANIZ	Строка (248)	Организация
RETURN_ID	Строка (255)	Направление
ISN_REPORT_EVENT	Целое число	Событие
EVENT_DATE	Дата и время	Время события
INFO	Текст	Информация
ISN_CITIZEN	Целое число	ISN гражданина
ISN_RESOLUTION	Целое число	ISN резолюции

SEV_REPORT_EVENT

Событие в докладе:

Таблица Г. 233

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN события
WEIGHT	Целое число	Вес элемента
CLASSIF_NAME	Строка (64)	Наименование события
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак логического удаления
NOTE	Строка (255)	Примечание

SEV_RULE

Правило СЭВ:

Таблица Г. 234

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN правил
CLASSIF_NAME	Строка (64)	Наименование правила МЭДО

Имя	Тип	Комментарии
RULE_KIND	Целое число	Вид правила
DUE_DOCGROUP	Строка (248)	Группа документов
FILTER_CONFIG	Текст	Условия
SCRIPT_CONFIG	Текст	Параметры обработки
WEIGHT	Целое число	Вес элемента
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак логического удаления
DUE_DEP	Строка (248)	Подразделение
NOTE	Строка (255)	Примечание
INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Когда добавил
UPD_WHO	Целое число	Кто изменил
UPD_DATE	Дата и время	Когда изменил

SEV_SUBSCRIBE

Таблица подписок СЭВ:

Таблица Г. 235

Имя	Тип	Комментарии
ISN_SUBSCRIBE	Целое число	ISN подписки
ISN_DOC	Целое число	ISN документа
KIND_DOC	Целое число	Вид документа
DUE_OWNER_ORGANIZ	Строка (248)	DUE организации адресата
PARTICIPANT_ISN	Целое число	ISN отправителя (участник СЭВ)
RETURN_ID	Строка (255)	Код возврата для уведомлений и докладов
EDMS_NAME	Строка (255)	Имя СЭД отправителя документа
EDMS_VERSION	Строка (64)	Номер версии СЭД отправителя документа

Имя	Тип	Комментарии
EDMS_UID	Строка (255)	UID СЭД отправителя документа
DOC_UID	Строка (255)	UID документа в системе отправителя
DOC_VERSION_UID	Строка (255)	UID версии проекта документа
DOC_NUMBER	Строка (255)	Номер документа
DOC_DATE	Дата и время	Дата документа
DOC_GROUP	Строка (64)	Наименование группы документов
DOC_GROUP_UID	Строка (255)	UID группы документов
SUBSCRIPTIONS	Строка (64)	Подписки из паспорта
HASH	Строка (255)	Хэш последнего доклада
TASKS_UIDS	Текст	Идентификаторы поручений
VISA_SIGN_UIDS	Текст	Идентификатор виз и подписей

SEV_SYNC_REPORT

Отчет о синхронизации СЭВ:

Таблица Г. 236

Имя	Тип	Комментарии
ISN_SEV_SYNC_REPORT	Целое число	ISN записи
ISN_PARTICIPANT	Целое число	ISN участника СЭВ
FILE_SYNC_DATE	Дата и время	Дата формирования файла синхронизации

SHABLON_DETAIL

Ограничения шаблонов номерообразования:

Таблица Г. 237

Имя	Тип	Комментарии
ISN_SHABLON_DETAIL	Целое число	ISN записи
DUE	Строка (248)	Код Дьюи группы
ELEMENT	Строка (2)	Элемент шаблона
CONSTR_TYPE	Строка (1)	тип ограничения

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN типа связи

SIGN_KIND_CL

Виды подписей (ЭП):

Таблица Г. 238

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN элемента
WEIGHT	Целое число	Вес элемента
CLASSIF_NAME	Строка (64)	Наименование
SIGN_TEXT	Строка (255)	Текст подписи
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак лог удаления

SMEV_PARTICIPANT

Участники СМЭВ:

Таблица Г. 239

Имя	Тип	Комментарии
ISN_SMEV_PARTICIPANT	Целое число	ISN участника СМЭВ
EXTERNAL_SYSTEM_CODE	Строка (255)	Код внешней системы
ISN_ORGANIZATION	Целое число	ISN организации участника отправитель запросов
CERTIFICATE_SP	Строка (255)	Сертификат ЭП СП
SIGN_ON_CLIENT	Целое число	Подписание ЭП СП на клиенте
AVAILABLE_PAGES	Целое число	Допустимые вкладки
DELETED	Целое число	Признак логического удаления

SMEV_SETTINGS_REQUEST

Настройки запросов СМЭВ:

Таблица Г. 240

Имя	Тип	Комментарии
ISN_SETTING_REQUEST	Целое число	Идентификатор настроек запроса
NAME	Строка (2000)	Название

Имя	Тип	Комментарии
REQUEST_NAMESPACE_URI	Строка (255)	Пространство имен вида сведений
REQUEST_TYPE_CODE	Строка (255)	Код запроса
ISN_ORGANIZATION	Целое число	ISN организации в адрес которой направляется запрос
DELETED	Целое число	Признак логического удаления

SMEV_SETTINGS_REQUEST_VERSION

Версионность настроек запроса СМЭВ:

Таблица Г. 241

Имя	Тип	Комментарии
ISN_SETTINGS_REQUEST	Целое число	Идентификатор настроек запроса
ISN_SMEV_PARTICIPANT	Целое число	Идентификатор записи участника СМЭВ
REQUEST_NAMESPACE_URI	Строка (255)	Пространство имён вида сведений

SRCH_AR_HIER

Перечень иерархий в поиске:

Таблица Г. 242

Имя	Тип	Комментарии
HIER_KIND	Целое число	Вид иерархии
HIER_NAME	Строка (255)	Название иерархии
HIER_USAGE	Строка (255)	Места использования

SRCH_CATEGORY

Категории осн реквизитов РК и РКПД в поиске:

Таблица Г. 243

Имя	Тип	Комментарии
ISN_NODE	Целое число	ISN категории
ISN_HIGH_NODE	Целое число	ISN вышестоящей вершины
LAYER	Целое число	Номер уровня
IS_NODE	Целое число	Признак вершины
WEIGHT	Целое число	Вес элемента

Имя	Тип	Комментарии
DUE	Строка (248)	Код DUE
KIND	Целое число	Вид критериев
CLASSIF_NAME	Строка (255)	Наименование категории
ISN_CRITERY	Целое число	ISN критерия
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак логического удаления

SRCH_CRITERY

Описания осн реквизитов РК и РКПД:

Таблица Г. 244

Имя	Тип	Комментарии
ISN_CRITERY	Целое число	ISN критерия
UOD	Строка (64)	Имя объекта
UOD_PROP	Строка (255)	Параметры объекта
TREE_LABEL	Строка (255)	Текст для отображения в дереве
FORM_LABEL	Строка (255)	Текст для отображения на форме
CRITERY_NAME	Строка (255)	Имя критерия для поисковой машины
KIND	Целое число	Вид критериев

SRCH_GROUP

Группы отчетов:

Таблица Г. 245

Имя	Тип	Комментарии
ISN_GROUP	Целое число	ISN записи
NAME	Строка (255)	Название
WEIGHT	Целое число	Вес
ALLOWED_FOR_ALL	Целое число	Разрешено всем
PROTECTED	Целое число	Защищенная запись
NOTE	Строка (2000)	Примечание

SRCH_GROUP_ITEM

Отчеты в группах:

Таблица Г. 246

Имя	Тип	Комментарии
ISN_GROUP	Целое число	ISN группы
ISN_REQUEST	Целое число	ISN отчета
WEIGHT	Целое число	Вес

SRCH_REQ_DESC

Описание сохраненного запроса:

Таблица Г. 247

Имя	Тип	Комментарии
ISN_REQUEST	Целое число	ISN сохраненного запроса
CRITERY_NAME	Строка (255)	Имя критерия для поисковой машины
VALUE	Строка (2000)	Значение
DISP_VALUE	Строка (2000)	Значение для отображения
RX_POS	Целое число	X реквизита на форме
RY_POS	Целое число	Y реквизита на форме
R_HEIGHT	Целое число	Высота реквизита на форме
R_WIDTH	Целое число	Ширина реквизита на форме
LX_POS	Целое число	X метки на форме
LY_POS	Целое число	Y метки на форме
L_HEIGHT	Целое число	Высота метки на форме
L_WIDTH	Целое число	Ширина метки на форме
TAB_ORDER	Целое число	Номер в порядке обхода

SRCH_REQUEST

Список сохраненных запросов:

Таблица Г. 248

Имя	Тип	Комментарии
ISN_REQUEST	Целое число	ISN запроса
REQUEST_NAME	Строка (255)	Название

Имя	Тип	Комментарии
SRCH_KIND_NAME	Строка (24)	Мнемоника вида поиска
PERSONAL	Целое число	"Личный запрос"
UTILITY_KIND	Целое число	Технологический
PROTECTED	Целое число	Признак запрета удаления
CUSTOMIZATION	Целое число	Настраиваемый
BASE_CONTROL	Строка (255)	Основной контрол
ISN_VIEW	Целое число	Представление
AUTO_DISTRIBUTE	Целое число	Автораздача пользователям
PARAMS	Текст	Параметры

SRCH_VIEW

Представление результатов запроса:

Таблица Г. 249

Имя	Тип	Комментарии
ISN_VIEW	Целое число	ISN представления
SRCH_KIND_NAME	Строка (24)	Тип списка
VIEW_NAME	Строка (255)	Название
ORDERBY	Строка (255)	Сортировка
SORT_DIRECTION	Целое число	Направление сортировки
PAGE_SIZE	Целое число	Размер страницы
PERSONAL	Целое число	Общий или личный
PROTECTED	Целое число	Признак запрета удаления
CUSTOMIZATION	Целое число	Настраиваемое
BASE_CONTROL	Строка (255)	Основное контрол
PARAMS	Текст	Настройки
AUTO_REFRESH	Целое число	Автоматическое обновление
HEIGHT	Целое число	Высота

SRCH_VIEW_DESC

Описание представления результатов запроса:

Таблица Г. 250

Имя	Тип	Комментарии
ISN_VIEW_DESC	Целое число	ISN записи
ISN_VIEW	Целое число	ISN представления
COLUMN_NUM	Целое число	Номер колонки
ROW_NUM	Целое число	Номер строки
ORDERNUM	Целое число	Порядковый номер
BLOCK_ID	Строка (255)	Блок
LABEL	Строка (255)	Заголовок
PARAMS	Текст	Параметры для блока

SSCAN_BARCODE

Хранение штрих кодов:

Таблица Г. 251

Имя	Тип	Комментарии
ISN_BARCODE	Целое число	Идентификатор записи
ISN_DOC	Целое число	Идентификатор документа
BARCODE_VALUE	Строка (255)	Штрих код
CREATED_AT	Дата и время	Дата и время создания записи

SSCAN_REQUEST

Заявка на УПС:

Таблица Г. 252

Имя	Тип	Комментарии
ISN_REQUEST	Целое число	Номер заявки
CREATED_AT	Дата и время	Дата и время создания
STATUS	Целое число	Статус заявки
ISN_USER	Целое число	Идентификатор пользователя
MRSCAN_ID	Глобальный уникальный идентификатор	Идентификатор заявки
ISN_BARCODE	Целое число	Идентификатор ШК

Имя	Тип	Комментарии
BARCODE_LOCATION	Целое число	Расположение ШК
PRINTER_TYPE	Целое число	Тип печатающего устройства
FIRST_PAGE_TYPE	Целое число	Тип первого листа с ШК РК
ORIENTATION_TYPE	Целое число	Ориентация листа
SAVE_FORMAT	Целое число	Формат сохранения
NEED_OCR	Логическое значение	Распознавание
NEED_VISUALITY	Логическое значение	Визуальный контроль
SCANNER_ALIAS	Строка (255)	Имя сканера
SCANNER_ID	Глобальный уникальный идентификатор	Идентификатор сканера
FILE_PROCESSING_RESULT	Целое число	Фиксация загрузки файлов в систему

SSTU_DECISION_HISTORY

История допов рубрик для выгрузки на ССТУ:

Таблица Г. 253

Имя	Тип	Комментарии
ISN_SSTU_DECISION_HISTORY	Целое число	ISN записи
ISN_DOC	Целое число	ISN документа
ISN_REF_RUBRIC	Целое число	ISN рубрики документа
REF_RUBRIC_VALUE	Строка (2000)	Значение рубрики
EOS_SSTU_STATUS	Строка (255)	Статус вопроса
EOS_SSTU_ACTIONS	Целое число	Приняты меры
EOS_SSTU_NO_REPLY	Дата и время	Оставлено без ответа
EOS_SSTU_ITEM	Целое число	Вопрос ССТУ
EOS_SSTU_ORGANIZ	Строка (255)	Рассмотрено в
EOS_SSTU_NEED_REPORT	Целое число	Включить в отчет
EOS_SSTU_SEND_REPORT	Строка (255)	Отчет отправлен
EOS_SSTU_BATCH	Строка (255)	Пакет выгрузки

Имя	Тип	Комментарии
EOS_SSTU_LINK	Строка (255)	Ссылка на РКИ
TRANSFER	Строка (255)	Организация
DEPARTMENT_ID	Строка (255)	ID организации
UPD_DATE	Дата и время	Дата создания изменения
UPD_WHO	Целое число	Кто редактировал
UPLOAD_SSTU_DATE	Дата и время	Дата и время формирования отчета
DOWNLOAD_SSTU_DATE	Дата и время	Дата и время загрузки информации на ССТУ

STATUS_EXEC_CL

Справочник Состояния исполнения (Контролера):

Таблица Г. 254

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN состояния исполнения
WEIGHT	Целое число	Вес элемента
CLASSIF_NAME	Строка (64)	Наименование
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак лог удаления
NOTE	Строка (255)	Комментарий
INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Когда добавил
UPD_WHO	Целое число	Кто изменил
UPD_DATE	Дата и время	Когда изменил

STATUS_REPLY_CL

Справочник Состояния исполнения (Исполнителя):

Таблица Г. 255

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN состояния исполнения
WEIGHT	Целое число	Вес элемента

Имя	Тип	Комментарии
CLASSIF_NAME	Строка (64)	Наименование
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак лог удаления
NOTE	Строка (255)	Комментарий
INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Когда добавил
UPD_WHO	Целое число	Кто изменил
UPD_DATE	Дата и время	Когда изменил

STOP_WORDS

Стоп слова:

Таблица Г. 256

Имя	Тип	Комментарии
WORD	Строка (64)	Стоп слово

STTEXT

Стандартный текст:

Таблица Г. 257

Имя	Тип	Комментарии
ISN_STTEXT	Целое число	ISN стандартного текста
ISN_STTEXT_LIST	Целое число	ISN списка стандартных текстов
CODE	Строка (24)	Код текста
STTEXT	Строка (2000)	Стандартный текст
WEIGHT	Целое число	Вес

STTEXT_CONTROL

Поле ввода стандартных текстов:

Таблица Г. 258

Имя	Тип	Комментарии
ISN_USER	Целое число	ISN пользователя
CONTROL_ID	Строка (255)	ID поля ввода

Имя	Тип	Комментарии
ISN_STTEXT_LIST	Целое число	ISN списка стандартных текстов

STTEXT_LIST

Список стандартных текстов:

Таблица Г. 259

Имя	Тип	Комментарии
ISN_STTEXT_LIST	Целое число	ISN списка стандартных текстов
ISN_USER	Целое число	ISN пользователя
LIST_NAME	Строка (255)	Наименование списка
REF_ISN_STTEXT_LIST	Целое число	Указатель на общий список стандартных текстов
ALL_FLAG	Целое число	Флаг копирования общих списков стандартных текстов
WEIGHT	Целое число	Вес

SW_LIST

Таблица Г. 260

Имя	Тип	Комментарии
ISN_SW_LIST	NUMBER(10)	Идентификатор выгружаемого файла
SW_FILENAME	VARCHAR2(255)	Название файла
VAR_NAME	VARCHAR2(255)	Параметр файла
PART_COUNT	NUMBER(4)	Порядковый номер выгрузки файла
VERSION	VARCHAR2(24)	Версия файла

SW_MODULES

Таблица Г. 261

Имя	Тип	Комментарии
ISN_SW_LIST	NUMBER(10)	Идентификатор выгружаемого файла
PART_NUMBER	NUMBER(4)	Порядковый номер блока
PART_CONTENTS	BLOB	Содержимое блока

SW_USE

Таблица Г. 262

Имя	Тип	Комментарии
ISN_SOFTWARE	NUMBER(10)	Идентификатор обновляемого приложения
ISN_SW_LIST	NUMBER(10)	Ссылка на список выгружаемых файлов

T_WORDS

Слова из текстов:

Таблица Г. 263

Имя	Тип	Комментарии
WORD	Строка (64)	Слово
WORD_ID	Целое число	Ид слова
WORD_CNT	Целое число	WORD_CNT

T_WU

Связь слов с документами:

Таблица Г. 264

Имя	Тип	Комментарии
WORD_ID	Целое число	Ид слова
ISN	Целое число	ISN документа

TEMP_RC

Временная РК:

Таблица Г. 265

Имя	Тип	Комментарии
ISN_TEMP_RC	Целое число	ISN РК
WAPI_SESSION_SID	Строка (255)	SID сессии
ISL_SID	Целое число	ISL сессии
OPERATION_KEY	Строка (255)	Ключ операции
EXPIRATION_DATE	Дата и время	Дата протухания
INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Когда добавил

UFOLDER

Личные папки пользователя:

Таблица Г. 266

Имя	Тип	Комментарии
ISN_REF_UFOLDER	Целое число	ISN папки
ISN_USER	Целое число	ISN пользователя
FOLDER_KIND	Целое число	Вид папки
FOLDER_NAME	Строка (64)	Наименование
COMMENTR	Строка (255)	Комментарий
QUANTITY	Целое число	Кол во документов
CREATE_DATE	Дата и время	Дата создания
EDIT_DATE	Дата и время	Дата модификации

UFOLDER_ACCESS

Доступ к личным папкам:

Таблица Г. 267

Имя	Тип	Комментарии
ISN_UFOLDER_ACCESS	Целое число	ISN записи
ISN_UFOLDER	Целое число	ISN папки
ISN_GRANTEE	Целое число	Пользователь
CAN_READ	Целое число	Право на чтение
CAN_UPDATE	Целое число	Право на изменение
CAN_DELETE	Целое число	Право на удаление
CAN_ADMIN	Целое число	Право на управление

USER_AUDIT

Аудит изменений справочника пользователей:

Таблица Г. 268

Имя	Тип	Комментарии
ISN_EVENT	Целое число	ISN события аудита
ISN_USER	Целое число	ISN редактируемого пользователя
ISN_WHO	Целое число	ISN редактирующего

		пользователя
EVENT_DATE	Дата и время	Дата события аудита
EVENT_KIND	Целое число	Вид события аудита

USER_AUTH_EXTERNAL_ID

Идентификаторы внешних провайдеров:

Таблица Г. 269

Имя	Тип	Комментарии
ISN_EXTERNAL_ID	Целое число	ISN идентификатора
ISN_LCLASSIF	Целое число	ISN пользователя
EXTERNAL_TYPE	Строка (50)	Тип идентификатора
EXTERNAL_ID	Строка (250)	ID идентификатора

USER_CABINET

Пользователи кабинета:

Таблица Г. 270

Имя	Тип	Комментарии
ISN_CABINET	Целое число	ISN кабинета
ISN_LCLASSIF	Целое число	ISN пользователя
FOLDERS_AVAILABLE	Строка (24)	Права на папки
ORDER_WORK	Целое число	Признак работы с поручениями
HOME_CABINET	Целое число	Признак гл кабинета
HIDE_INACCESSIBLE	Целое число	"Скрыть недоступные"
HIDE_INACCESSIBLE_PRJ	Целое число	"Скрыть недоступные" для РКПД
IS_ASSISTANT	Целое число	Является помощников владельца кабинета

USER_CARD_DOCGROUP

Группы документов пользователя:

Таблица Г. 271

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN пользователя
DUE_CARD	Строка (248)	Код Дьюи ДЛ_картотеки

DUE	Строка (248)	Код Дьюи группы док
FUNC_NUM	Целое число	Позиция права в маске
ALLOWED	Целое число	Разрешение

USER_CERT_PROFILE

Профили сертификатов пользователей:

Таблица Г. 272

Имя	Тип	Комментарии
ISN_CERT_PROFILE	Целое число	ISN профиля сертификатов пользователей
ISN_USER	Целое число	Пользователь
ID_CERTIFICATE	Текст	ID сертификата

USER_CERTIFICATE

Сертификаты пользователей:

Таблица Г. 273

Имя	Тип	Комментарии
ISN_USER	Целое число	ISN пользователя
SIGN_CERT	Текст	Сертификат подписи
ENC_CERT	Текст	Сертификат шифрования
SIGN_MAIL_CERT	Текст	Сертификат подписи email
ENC_MAIL_CERT	Текст	Сертификат шифрования email

USER_CL

Справочник Пользователей:

Таблица Г. 274

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN пользователя
SECURLEVEL	Целое число	Гриф доступа
DUE_DEP	Строка (248)	Должностное лицо
WEIGHT	Целое число	Вес элемента
SURNAME_PATRON	Строка (64)	Имя пользователя
PROTECTED	Целое число	Пр запрета удаления

Имя	Тип	Комментарии
DELETED	Целое число	Пр лог удаления
CLASSIF_NAME	Строка (256)	Идентификатор
PASSWORD	Строка (64)	Пароль
NOTE	Строка (255)	Наименование подразделения
ORACLE_ID	Строка (255)	Ид ORACLE
ADMIN	Целое число	Пр Администратора
USERTYPE	Целое число	Тип пользователя
AV_SYSTEMS	Строка (64)	Доступные системы
ADM_SYSTEMS	Строка (20)	Администратор систем
DELO_RIGHTS	Строка (40)	Права в ДЕЛЕ абсолютные
STREAM_SCAN_RIGHTS	Строка (20)	Права в поточном ск
ARCHIVE_RIGTHS	Строка (20)	Права в архиве
TECH_RIGHTS	Строка (64)	Справочники доступные системному технолог
LOGIN_ATTEMPTS	Целое число	Было неудачных попыток входа
IS_PASSWORD	Целое число	Проверенность пароля
IS_SECUR_ADM	Целое число	Администратор системы
PASSWORD_DATE	Дата и время	Дата смены пароля
TECH_DUE_DEP	Строка (248)	Подразделение пользователя
NOTE2	Строка (2000)	Примечание

USER_DOCGROUP_ACCESS

Доступ пользователей к группам документов:

Таблица Г. 275

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN пользователя
DUE	Строка (248)	Код Дьюи группы
ALLOWED	Целое число	Разрешения доступа

USER_EDIT_ORG_TYPE

Права пользователей на типы организаций:

Таблица Г. 276

Имя	Тип	Комментарии
ISN_USER	Целое число	ISN пользователя
ISN_ORG_TYPE	Целое число	ISN типа организации

USER_FILESECUR

Гриффы доступа пользователей к файлам:

Таблица Г. 277

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN пользователя
SECURLEVEL	Целое число	Гриф файла

USER_HISTORY

История работы пользователя:

Таблица Г. 278

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN пользователя
OPERATION	Строка (255)	Код операции
ISN_OBJ	Целое число	ISN объекта
KIND_OBJ	Целое число	Ссылка на ISN_EOS_OBJECT
LAST_ACCESS_TIME	Дата и время	Время последнего выполнения операции

USER_LISTS

Списки пользователя:

Таблица Г. 279

Имя	Тип	Комментарии
ISN_LIST	Целое число	isn_list
CLASSIF_ID	Целое число	Ид классификатора
ISN_LCLASSIF	Целое число	ISN пользователя
NAME	Строка (64)	Имя списка
WEIGHT	Целое число	Вес элемента

REF_ISN_LIST	Целое число	Указатель на общий список
ALL_FLAG	Целое число	Флаг копирования общих списков
LIST_KIND	Целое число	Вид списка
CAN_EDIT	Целое число	Право редактирования
CAN_DELETE	Целое число	Право удаления из списка

USER_ORGANIZ

Права ведения внешних резолюций для организаций:

Таблица Г. 280

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN пользователя
DUE	Строка (248)	Код Дьюи организации
FUNC_NUM	Целое число	Номер функции
WEIGHT	Целое число	Вес

USER_PARMs

Настройки (параметры) пользователя:

Таблица Г. 281

Имя	Тип	Комментарии
ISN_USER_OWNER	Целое число	пользователь
PARM_NAME	Строка (64)	имя параметра
PARM_GROUP	Целое число	группа параметра
PARM_VALUE	Строка (2000)	значение параметра

USER_REQUEST

Сохраненные запросы пользователей:

Таблица Г. 282

Имя	Тип	Комментарии
ISN_USER	Целое число	ISN пользователя
ISN_REQUEST	Целое число	ISN запроса
WEIGHT	Целое число	Вес
PARAMS	Текст	Параметры папок кабинетов

USER_RIGHT_DOCGROUP

Ограничения прав пользователей по группам документов:

Таблица Г. 283

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN пользователя
FUNC_NUM	Целое число	номер права
DUE	Строка (248)	DUE группы документов
ALLOWED	Целое число	Разрешено

USER_SETTINGS

Настройки пользователей:

Таблица Г. 284

Имя	Тип	Комментарии
ID	Целое число	Идентификатор
USER_ID	Целое число	Пользователь
MODULE	Строка (64)	Модуль
CATEGORY	Строка (64)	Категория
SUBCATEGORY	Строка (64)	Подкатегория
SETTINGS_KEY	Строка (256)	Ключ настройки
DATA_TYPE	Целое число	Тип данных
VALUE	Строка (512)	Значение
TIME_STAMP	Дата и время	Дата время

USER_SRCH_GROUP

Права пользователей на группы отчетов:

Таблица Г. 285

Имя	Тип	Комментарии
ISN_USER	Целое число	ISN пользователя
ISN_GROUP	Целое число	ISN группы отчетов

USER_TECH

Права пользователей на справочниках:

Таблица Г. 286

Имя	Тип	Комментарии
-----	-----	-------------

ISN_LCLASSIF	Целое число	ISN пользователя
FUNC_NUM	Целое число	Номер права
DUE	Строка (248)	Дюе
CLASSIF_ID	Целое число	ID справочника
ALLOWED	Целое число	Разрешение

USER_VIEW

Представление пользователя результатов запроса:

Таблица Г. 287

Имя	Тип	Комментарии
ISN_USER	Целое число	ISN пользователя
ISN_VIEW	Целое число	ISN представления
WEIGHT	Целое число	Вес

USERCARD

Картотечные права пользователя:

Таблица Г. 288

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN пользователя
DUE	Строка (248)	Код Дьюи ДЛ_картотеки
HOME_CARD	Целое число	Признак гл картотеки
FUNCLIST	Строка (24)	Перечень функций

USERDEP

Перечни ДЛ для абсолютных прав

Таблица Г. 289

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN пользователя
FUNC_NUM	Целое число	func_num
DUE	Строка (248)	Код Дьюи ДЛ
WEIGHT	Целое число	Вес
DEEP	Целое число	Учитывать вложенные
ALLOWED	Целое число	Разрешено

USERSECUR

Гриффы доступа пользователя:

Таблица Г. 290

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN пользователя
SECURLEVEL	Целое число	Гриф доступа

VIEWPROT

Протокол просмотра:

Таблица Г. 291

Имя	Тип	Комментарии
TABLE_ID	Строка (1)	Идентификатор таблицы
OPER_ID	Строка (1)	Идентификатор операции
SUBOPER_ID	Строка (1)	Идентификатор подоперации
OPER_DESCRIBE	Строка (3)	Описатель операции
REF_ISN	Целое число	ISN объекта
TIME_STAMP	Дата и время	Временная метка
USER_ISN	Целое число	ISN пользователя
OPER_COMMENT	Строка (255)	Комментарий к операции

VISA_TYPE_CL

Справочник Типы визы:

Таблица Г. 292

Имя	Тип	Комментарии
ISN_LCLASSIF	Целое число	ISN типа визы
WEIGHT	Целое число	Вес элемента
CLASSIF_NAME	Строка (64)	Наименование
PROTECTED	Целое число	Признак запрета удаления
DELETED	Целое число	Признак лог удаления
NOTE	Строка (255)	Комментарий
IS_FINAL	Целое число	Признак финальной визы
STATUS	Строка (24)	Статус визы

INS_WHO	Целое число	Кто добавил
INS_DATE	Дата и время	Когда добавил
UPD_WHO	Целое число	Кто изменил
UPD_DATE	Дата и время	Когда изменил

WAPI_SESSION

Сессии из Web приложений:

Таблица Г. 293

Имя	Тип	Комментарии
ISL_SID	Целое число	ISL сессии web-приложения
SID	Строка (255)	Идентификатор сессии в cookie
SESSION_ATTACH_TOKEN	Строка (255)	Токен подключения к сессии
UISN	Целое число	ISN пользователя
LOGIN_TIME	Дата и время	Время входа в систему
LAST_REQUEST_TIME	Дата и время	Время последнего запроса
EXPIRATION_TIME	Дата и время	Срок истечения сессии при неактивности
DB_SPID	Строка (255)	Идентификатор сессии СУБД (не используется)
AUTH_TICKET	VARBINARY (2000)	Токен авторизации внешней системы (не используется)
APP	Строка (64)	Приложение

WAPI_SESSION_STORE

Хранение данных веб сессий:

Таблица Г. 294

Имя	Тип	Комментарии
ISL_SID	Целое число	ISL сессии web-приложения
VALUE_ID	Строка (255)	Идентификатор значения
VALUE	Текст	Значение

WEIGHT_DUE

Весовые коды DUE (служебная таблица):

Таблица Г. 295

Имя	Тип	Комментарии
DUE	Строка (248)	Код DUE записи справочника
CLS	Строка (30)	Справочник
WDUE	Строка (250)	Весовой код записи справочника

Приложение Д

Перечень связей таблиц системы «ДЕЛО»

Перечень связей таблиц системы «ДЕЛО» представлен ниже (см. Таблица Д. 1).

Таблица Д. 1

Главная таблица	Связанная таблица	Первичный ключ	Внешний ключ
ADDR_CATEGORY_CL	ORGANIZ_CL	ISN_LCLASSIF	ISN_ADDR_CATEGORY
ADDR_CATEGORY_CL	CITIZEN	ISN_LCLASSIF	ISN_ADDR_CATEGORY
ADDRESS	REF_SEND	ISN_ADDRESS	ISN_ADDRESS
ADDRESS	SEND_PACKAGE	ISN_ADDRESS	ISN_ADDRESS
ADDRESS_VID_CL	ADDRESS	ISN_LCLASSIF	ISN_ADDRESS_VID
APP_HOST_CONFIG	APP_HOST_PULSE	ISN_APP_HOST_CONFIG	ISN_APP_HOST_CONFIG
APP_HOST_CONFIG	APP_HOST_CONFIG_DAEMON_TYPE	ISN_APP_HOST_CONFIG	ISN_APP_HOST_CONFIG
APP_HOST_INSTANCE	APP_HOST_CONFIG	ISN_APP_HOST_INSTANCE	ISN_APP_HOST_INSTANCE
APP_HOST_INSTANCE	DAEMON_INSTANCE_BIN	ISN_APP_HOST_INSTANCE	ISN_APP_HOST_INSTANCE
APP_HOST_VIRTUAL	APP_HOST_ASSIGNED_INSTANCE	ISN_APP_HOST_VIRTUAL	ISN_APP_HOST_VIRTUAL
AR_CLS_CONTROL	AR_DESCRIPTOR	ISN_CONTROL	ISN_CLS_CONTROL
AR_DESCRIPTOR	AR_VALUE_LIST	ISN_AR_DESCRIPTOR	ISN_AR_DESCRIPTOR
AR_DESCRIPTOR	AR_DOCGROUP	ISN_AR_DESCRIPTOR	ISN_AR_DESCRIPTOR
AR_DESCRIPTOR	AR_CATEGORY	ISN_AR_DESCRIPTOR	ISN_AR_DESCRIPTOR
BPM_COMPONENT_TYPE	BPM_COMP_PROCESS_TYPE	ISN_COMPONENT_TYPE	ISN_COMPONENT_TYPE
BPM_COMPONENT_TYPE	BPM_PROC_VER_COMPONENT	ISN_COMPONENT_TYPE	ISN_COMPONENT_TYPE
BPM_COMPONENT_TYPE	BPM_INSTANCE_COMPONENT	ISN_COMPONENT_TYPE	ISN_COMPONENT_TYPE
BPM_INSTANCE	BPM_INSTANCE_COMPONENT	ISN_INSTANCE	ISN_INSTANCE

Главная таблица	Связанная таблица	Первичный ключ	Внешний ключ
BPM_INSTANCE_COMPONENT_EXEC	PROCESS_TASK_COMPETITOR	ISN_INSTANCE_COMPONENT_EXEC	ISN_INSTANCE_COMPONENT_EXEC
BPM_INSTANCE_COMPONENT_PARAM	BPM_INSTANCE_COMPONENT_EXEC	ISN_INSTANCE_COMPONENT_PARAM	ISN_INSTANCE_COMPONENT_PARAM
BPM_INSTANCE_COMPONENT_PARAM	BPM_INSTANCE_COMPONENT_REP	ISN_INSTANCE_COMPONENT_PARAM	ISN_INSTANCE_COMPONENT_PARAM
BPM_INSTANCE_COMPONENT_PARAM	BPM_INSTANCE_COMPONENT_ENDORSE	ISN_INSTANCE_COMPONENT_PARAM	ISN_INSTANCE_COMPONENT_PARAM
BPM_INSTANCE_COMPONENT_PARAM	BPM_INSTANCE_COMPONENT_SIGN	ISN_INSTANCE_COMPONENT_PARAM	ISN_INSTANCE_COMPONENT_PARAM
BPM_INSTANCE_COMPONENT	BPM_INSTANCE_COMPONENT_PARAM	ISN_INSTANCE_COMPONENT	ISN_INSTANCE_COMPONENT
BPM_INSTANCE_COMPONENT	PROCESS_TASK	ISN_INSTANCE_COMPONENT	ISN_INSTANCE_COMPONENT
BPM_PROC_VERSION_COMPONENT_PARAM	BPM_PROC_VERSION_COMPONENT_EXEC	ISN_PROC_VERSION_COMPONENT_PARAM	ISN_PROC_VERSION_COMPONENT_PARAM
BPM_PROC_VERSION_COMPONENT_PARAM	BPM_PROC_VERSION_COMPONENT_REP	ISN_PROC_VERSION_COMPONENT_PARAM	ISN_PROC_VERSION_COMPONENT_PARAM
BPM_PROC_VERSION_COMPONENT_PARAM	BPM_PROC_VERSION_COMPONENT_ENDORSE	ISN_PROC_VERSION_COMPONENT_PARAM	ISN_PROC_VERSION_COMPONENT_PARAM
BPM_PROC_VERSION_COMPONENT_PARAM	BPM_PROC_VERSION_COMPONENT_SIGN	ISN_PROC_VERSION_COMPONENT_PARAM	ISN_PROC_VERSION_COMPONENT_PARAM
BPM_PROC_VERSION_COMPONENT	BPM_PROC_VERSION_COMPONENT_PARAM	ISN_PROC_VERSION_COMPONENT	ISN_PROC_VERSION_COMPONENT
BPM_PROC_VERSION_COMPONENT	BPM_INSTANCE_COMPONENT	ISN_PROC_VERSION_COMPONENT	ISN_PROC_VERSION_COMPONENT
BPM_PROCESS	BPM_PROCESS_VERSION	ISN_PROCESS	ISN_PROCESS
BPM_PROCESS_ROLE	BPM_PROC_VERSION_COMPONENT_EXEC	ISN_LCLASSIF	ISN_ROLE_EXEC
BPM_PROCESS_ROLE	BPM_INSTANCE_COMPONENT_EXEC	ISN_LCLASSIF	ISN_ROLE_EXEC
BPM_PROCESS_TYPE	BPM_PROCESS	ISN_PROCESS_TYPE	ISN_PROCESS_TYPE
BPM_PROCESS_TYPE	BPM_COMP_PROCESS_TYPE	ISN_PROCESS_TYPE	ISN_PROCESS_TYPE
BPM_PROCESS_TYPE	BPM_USER_PROCESS_TYPE	ISN_PROCESS_TYPE	ISN_PROCESS_TYPE

Главная таблица	Связанная таблица	Первичный ключ	Внешний ключ
BPM_PROCESS_VERSION	BPM_PROC_VER_DG	ISN_PROCESS_VERSION	ISN_PROCESS_VERSION
BPM_PROCESS_VERSION	BPM_PROC_VER_PRJ_STAGE	ISN_PROCESS_VERSION	ISN_PROCESS_VERSION
BPM_PROCESS_VERSION	BPM_PROC_VER_PRJ	ISN_PROCESS_VERSION	ISN_PROCESS_VERSION
BPM_PROCESS_VERSION	BPM_PROC_VER_SCHEDULE	ISN_PROCESS_VERSION	ISN_PROCESS_VERSION
BPM_PROCESS_VERSION	BPM_PROC_VER_OBJECT	ISN_PROCESS_VERSION	ISN_PROCESS_VERSION
BPM_PROCESS_VERSION	BPM_INSTANCE	ISN_PROCESS_VERSION	ISN_PROCESS_VERSION
BPM_PROCESS_VERSION	BPM_PROC_VER_COMPONENT	ISN_PROCESS_VERSION	ISN_PROCESS_VERSION
BUF_FOLDER	BUF_MESSAGE	ISN_FOLDER	ISN_FOLDER
BUF_FOLDER	BUF_RULE	ISN_FOLDER	ISN_FOLDER
BUF_FOLDER	BUF_USER_FOLDER	ISN_FOLDER	ISN_FOLDER
BUF_MESSAGE	BUF_MESSAGE_FIELDS	ISN_MESSAGE	ISN_MESSAGE
CABINET	USER_CABINET	ISN_CABINET	ISN_CABINET
CABINET	FOLDER	ISN_CABINET	ISN_CABINET
CABINET	NEW_RECORD_CABINET	ISN_CABINET	ISN_CABINET
CITIZEN	REF_LETTER	ISN_CITIZEN	ISN_CITIZEN
CITIZEN	REF_SEND	ISN_CITIZEN	ISN_CITIZEN
CITIZEN	CITIZEN_STATUS	ISN_CITIZEN	ISN_CITIZEN
CITIZEN	PRJ_REF_SEND	ISN_CITIZEN	ISN_CITIZEN
CITIZEN	AR_CITIZEN_VALUE	ISN_CITIZEN	ISN_CITIZEN
CITSTATUS_CL	CITIZEN_STATUS	ISN_NODE	ISN_STATUS
CONTACT	DEPARTMENT	ISN_CONTACT	ISN_CONTACT
CONTACT	REF_SEND	ISN_CONTACT	ISN_CONTACT
CONTACT	REF_CORRESP	ISN_CONTACT	ISN_CONTACT

Главная таблица	Связанная таблица	Первичный ключ	Внешний ключ
CONTACT	PRJ_REF_SEND	ISN_CONTACT	ISN_CONTACT
CONTACT	REF_SOISP	ISN_CONTACT	ISN_CONTACT
DAEMON_GROUP	DAEMON_TYPE	ISN_DAEMON_GROUP	ISN_DAEMON_GROUP
DAEMON_INSTANCE	DAEMON_INSTANCE_BIND	ISN_DAEMON_INSTANCE	ISN_DAEMON_INSTANCE
DAEMON_TYPE	DAEMON_TYPE_VERSION	ISN_DAEMON_TYPE	ISN_DAEMON_TYPE
DAEMON_TYPE	DAEMON_INSTANCE	ISN_DAEMON_TYPE	ISN_DAEMON_TYPE
DAEMON_TYPE_VERSION	APP_HOST_CONFIG_DAEMON_TYPE	ISN_DAEMON_TYPE_VERSION	ISN_DAEMON_TYPE_VERSION
DELIVERY_CL	REF_SEND	ISN_LCLASSIF	ISN_DELIVERY
DELIVERY_CL	REESTRTYPE_CL	ISN_LCLASSIF	ISN_DELIVERY
DELIVERY_CL	DOC_RC	ISN_LCLASSIF	ISN_DELIVERY
DELO_BLOB	DEPARTMENT	ISN_BLOB	ISN_PHOTO
DEPARTMENT	RESOLUTION	DUE	DUE
DEPARTMENT	REPLY	DUE	DUE
DEPARTMENT	REF_ACCESS_CARD	DUE	DUE
DEPARTMENT	REF_VISA	DUE	DUE_PERSON
DEPARTMENT	JOURNAL	DUE	DUE_CORRESP
DEPARTMENT	JOURNAL	DUE	DUE_ADRESAT
DEPARTMENT	ACQUAINTANCE	DUE	DUE_ADRESAT
DEPARTMENT	USERDEP	DUE	DUE
DEPARTMENT	PRJ_VISA_SIGN	DUE	DUE_PERSON
DEPARTMENT	PRJ_VISA_SIGN	DUE	DUE_PERSON_PARENT
DEPARTMENT	PRJ_REF_SEND	DUE	DUE_DEP
DEPARTMENT	DOC_RC	DUE	DUE_PERSON_EXE
DEPARTMENT	DOC_WHO	DUE	DUE_PERSON

Главная таблица	Связанная таблица	Первичный ключ	Внешний ключ
DEPARTMENT	DOC_SIGN	DUE	DUE_PERSON
DEPARTMENT	REF_FILE_EDS	DUE	DUE_PERSON
DEPARTMENT	RESOLUTION	DUE	DUE_CONTROLLER
DEPARTMENT	REF_FILE_ACCESS	DUE	DUE_DEP
DEPARTMENT	RESOLUTION_CARD	DUE	DUE
DEPARTMENT	PRJ_EXEC	DUE	DUE_PERSON
DEPARTMENT	NTFY_EVENT	DUE	DUE_PERSON
DEPARTMENT	DEPARTMENT	ISN_NODE	ISN_HIGH_NODE
DEPARTMENT	DOC_EXE	DUE	DUE_PERSON
DEPARTMENT	SEV_RULE	DUE	DUE_DEP
DEPARTMENT	BUF_RULE	DUE	DUE_DEP
DEPARTMENT	DEPARTMENT_REPL	DUE	DUE_REPL
DEPARTMENT	DEPARTMENT_REPL	DUE	DUE_DEP
DEPARTMENT	DEP_REPLACE	DUE	DUE
DEPARTMENT	FORUM_DUE	DUE	DUE_DEP
DEPARTMENT	PROCESS_TASK	ISN_NODE	ISN_EXEC
DEPARTMENT	PROCESS_TASK_COMPETITOR	ISN_NODE	ISN_EXEC
DEPARTMENT	BPM_PROCESS_ROLE	ISN_NODE	ISN_PERSON
DEPARTMENT	BPM_PROC_VER_COMP_EXEC	ISN_NODE	ISN_EXEC
DEPARTMENT	BPM_INSTANCE_COMP_EXEC	ISN_NODE	ISN_EXEC
DEPARTMENT	REF_SEND	DUE	DUE_SENDING_DEP
DEPARTMENT	MREVT_ASSOCIATION	DUE	DUE_DEPARTMENT
DOC_DEFAULT	DOC_DEFAULT_VALUE	DEFAULT_ID	DEFAULT_ID
DOC_RC	REF_CORRESP	ISN_DOC	ISN_DOC_INP
DOC_RC	REF_LETTER	ISN_DOC	ISN_DOC_INP

Главная таблица	Связанная таблица	Первичный ключ	Внешний ключ
DOC_RC	REF_VISA	ISN_DOC	ISN_DOC_OUT
DOC_RC	REF_SOISP	ISN_DOC	ISN_DOC_OUT
DOC_RC	DOC_FOLDER_ITEM	ISN_DOC	ISN_DOC
DOC_RC	FORWARD	ISN_DOC	ISN_DOC
DOC_RC	JOURNAL	ISN_DOC	ISN_DOC
DOC_RC	REF_ACCESS_CARD	ISN_DOC	ISN_REF_DOC
DOC_RC	ACQUAINTANCE	ISN_DOC	ISN_DOC
DOC_RC	REF_SEND	ISN_DOC	ISN_REF_DOC
DOC_RC	RESOLUTION	ISN_DOC	ISN_REF_DOC
DOC_RC	AR_RC_VALUE	ISN_DOC	ISN_RC
DOC_RC	DOC_WHO	ISN_DOC	ISN_DOC
DOC_RC	DOC_SIGN	ISN_DOC	ISN_DOC
DOC_RC	REF_RUBRIC	ISN_DOC	ISN_REF_DOC
DOC_RC	SEV_REPORT	ISN_DOC	ISN_DOC
DOC_RC	DOC_EXE	ISN_DOC	ISN_DOC
DOC_RC	HELPER_T1	ISN_DOC	ISN
DOC_RC	DOC_ORGANIZ_EXCLUDE	ISN_DOC	ISN_DOC
DOC_RC	SSTU_DECISION_HISTORY	ISN_DOC	ISN_DOC
DOCGROUP_CL	USER_CARD_DOCGROUP	DUE	DUE
DOCGROUP_CL	SHABLON_DETAIL	DUE	DUE
DOCGROUP_CL	PRJ_RC	DUE	DUE_DOCGROUP
DOCGROUP_CL	DOC_RC	DUE	DUE_DOCGROUP
DOCGROUP_CL	USER_DOCGROUP_ACCESS	DUE	DUE
DOCGROUP_CL	DOC_DEFAULT_VALUE	ISN_NODE	ISN_DOCGROUP
DOCGROUP_CL	PRJ_DEFAULT_VALUE	ISN_NODE	ISN_DOCGROUP

Главная таблица	Связанная таблица	Первичный ключ	Внешний ключ
DOCGROUP_CL	SEV_RULE	DUE	DUE_DOCGROUP
DOCGROUP_CL	USER_RIGHT_DOCGROUP	DUE	DUE
DOCGROUP_CL	DG_FILE_CATEGORY	ISN_NODE	ISN_NODE_DG
DOCGROUP_CL	BPM_PROC_VER_DG	DUE	DUE_DOCGROUP
DOCVID_CL	DOC_RC	ISN_LCLASSIF	ISN_DOCVID
DOCVID_CL	DOCGROUP_CL	ISN_LCLASSIF	ISN_DOCVID
EDS_CATEGORY_CL	CA_CATEGORY	ISN_LCLASSIF	ISN_EDS_CATEGOR Y
EVNT_SUBSCRIPTION	EVNT_QUEUE_ITEM	ISN_SUBSCRIPTION	ISN_SUBSCRIPTION
FILE_CATEGORY_CL	DG_FILE_CATEGORY	ISN_LCLASSIF	ISN_FILE_CATEGOR Y
FILE_CATEGORY_CL	REF_FILE	ISN_LCLASSIF	ISN_FILE_CATEGOR Y
FILE_TYPE_CL	REF_FILE	ISN_LCLASSIF	ISN_FILE_TYPE
FOLDER	DOC_FOLDER_ITEM	ISN_FOLDER	ISN_FOLDER
FOLDER	PRJ_FOLDER_ITEM	ISN_FOLDER	ISN_FOLDER
FORUM_THEME	FORUM_MSG	ISN_FORUM_THEME	ISN_FORUM_THEME
FORUM_THEME	FORUM_DUE	ISN_FORUM_THEME	ISN_FORUM_THEME
FORUM_THEME	FORUM_READ	ISN_FORUM_THEME	ISN_FORUM_THEME
LIB_LIBRARY	LIB_FILELINK	ISN_LIBRARY	ISN_LIBRARY
LIB_LIBRARY	LIB_PARAM	ISN_LIBRARY	ISN_LIBRARY
LINK_CL	REF_LINK	ISN_LCLASSIF	ISN_CLLINK
LINK_CL	SHABLON_DETAIL	ISN_LCLASSIF	ISN_LCLASSIF
MEDO_NODE_CL	MEDO_PARTICIPANT	ISN_LCLASSIF	ISN_MAIN_NODE
MEDO_NODE_CL	MEDO_PARTICIPANT	ISN_LCLASSIF	ISN_DSP_NODE
MREVT_EVENT	MREVT_ASSOCIATION	ISN_EVENT	ISN_EVENT
MREVT_EVENT	MREVT_REF	ISN_EVENT	ISN_EVENT
NOMENKL_CL	DOC_RC	ISN_LCLASSIF	ISN_NOMENC

Главная таблица	Связанная таблица	Первичный ключ	Внешний ключ
NOMENKL_CL	PRJ_RC	ISN_LCLASSIF	ISN_NOMENKL
NTFY_OPERATION	NTFY_SUBSCRIPTION	CODE	OPERATION_CODE
NTFY_OPERATION	NTFY_EVENT	CODE	OPERATION_CODE
NTFY_OPERATION	NTFY_OPERATION	CODE	PARENT_CODE
NTFY_PERIODICITY	NTFY_SUBSCRIPTION	ISN_PERIODICITY	ISN_PERIODICITY
NTFYPF_CHANNEL	NTFYPF_MESSAGE	NTFYPF_CHANNEL_ID	NTFYPF_CHANNEL_ID
NTFYPF_MESSAGE	NTFYPF_MESSAGE_JOURNAL	NTFYPF_MESSAGE_ID	NTFYPF_MESSAGE_ID
ORG_TYPE_CL	ORGANIZ_CL	ISN_LCLASSIF	ISN_ORGANIZ_TYPE
ORG_TYPE_CL	USER_EDIT_ORG_TYPE	ISN_LCLASSIF	ISN_ORG_TYPE
ORGANIZ_CL	REF_SOISP	DUE	DUE_ORGANIZ
ORGANIZ_CL	PRJ_REF_SEND	DUE	DUE_ORGANIZ
ORGANIZ_CL	CONTACT	ISN_NODE	ISN_ORGANIZ
ORGANIZ_CL	REF_CORRESP	DUE	DUE_ORGANIZ
ORGANIZ_CL	REF_SEND	DUE	DUE_ORGANIZ
ORGANIZ_CL	USER_ORGANIZ	DUE	DUE
ORGANIZ_CL	SEV_PARTICIPANT	DUE	DUE_ORGANIZ
ORGANIZ_CL	SEV_REPORT	DUE	DUE_ORGANIZ
ORGANIZ_CL	ORGANIZ_CL	ISN_NODE	ISN_HIGH_NODE
ORGANIZ_CL	AR_ORGANIZ_VALUE	ISN_NODE	ISN_NODE
ORGANIZ_CL	DOC_ORGANIZ_EXCLUDE	DUE	DUE_LINK_ORGANIZ
ORGANIZ_CL	SMEV_PARTICIPANT	ISN_NODE	ISN_ORGANIZATION
ORGANIZ_CL	SMEV_SETTINGS_REQUEST	ISN_NODE	ISN_ORGANIZATION
PRINT_FORM	PRINT_FILE_FORMAT	ID	PRINT_FORM_ID
PRINT_FORM_CATEGORY	PRINT_FORM	PRINT_FORM_CATEGORY_ID	PRINT_FORM_CATEGORY_ID

Главная таблица	Связанная таблица	Первичный ключ	Внешний ключ
PRJ_DEFAULT	PRJ_DEFAULT_VALUE	DEFAULT_ID	DEFAULT_ID
PRJ_EXEC	PRJ_FOLDER_ITEM	ISN_PRJ_EXEC	ISN_PRJ_EXEC
PRJ_EXEC	PRJ_PREP_TASK	ISN_PRJ_EXEC	ISN_PRJ_EXEC
PRJ_RC	PRJ_FOLDER_ITEM	ISN_PRJ	ISN_PRJ
PRJ_RC	PRJ_REF_SEND	ISN_PRJ	ISN_REF_DOC
PRJ_RC	PRJ_VISA_SIGN	ISN_PRJ	ISN_PRJ
PRJ_RC	PRJ_EXEC	ISN_PRJ	ISN_PRJ
PRJ_RC	AR_PRJ_VALUE	ISN_PRJ	ISN_PRJ
PRJ_RC	PRJ_REF_RUBRIC	ISN_PRJ	ISN_REF_PRJ
PRJ_RC	PRJ_FORUM	ISN_PRJ	ISN_PRJ
PRJ_RC	PRJ_RETURN	ISN_PRJ	ISN_PRJ
PRJ_STAGE_CL	PRJ_RC	ISN_LCLASSIF	ISN_PRJ_STAGE
PRJ_STAGE_CL	BPM_PROC_VER_PRJ_STAGE	ISN_LCLASSIF	ISN_PRJ_STAGE
PRJ_VISA_SIGN	PRJ_FOLDER_ITEM	ISN_PRJ_VISA_SIGN	ISN_PRJ_VISA_SIGN
PROCESS_TASK	PROCESS_TASK	ISN_PROCESS_TASK	ISN_PARENT_TASK
PROCESS_TASK	PROCESS_TASK_COMPETITOR	ISN_PROCESS_TASK	ISN_PROCESS_TASK
PROCESS_TASK	PRJ_PREP_TASK	ISN_PROCESS_TASK	ISN_PROCESS_TASK
PROCESS_TASK	PRJ_ENDORSE_TASK	ISN_PROCESS_TASK	ISN_PROCESS_TASK
PROCESS_TASK	PRJ_SIGN_TASK	ISN_PROCESS_TASK	ISN_PROCESS_TASK
PROCESS_TASK_STATUS_CL	PROCESS_TASK	ISN_LCLASSIF	ISN_TASK_STATUS
PROT_STREAM_SCAN	PROT_FILE_SSCAN	ISN_PROT	ISN_PROT
REESTR_NEW	SEND_PACKAGE	ISN_REESTR	ISN_REESTR
REF_FILE	FILE_CONTENTS	ISN_REF_FILE	ISN_REF_FILE
REF_FILE	REF_FILE_EDS	ISN_REF_FILE	ISN_REF_FILE
REF_FILE	REF_FILE_ACCESS	ISN_REF_FILE	ISN_REF_FILE

Главная таблица	Связанная таблица	Первичный ключ	Внешний ключ
REF_RUBRIC	AR_RUBRIC_VALUE	ISN_REF_RUBRIC	ISN_REF_RUBRIC
REF_RUBRIC	SSTU_DECISION_HISTORY	ISN_REF_RUBRIC	ISN_REF_RUBRIC
REGION_CL	ORGANIZ_CL	ISN_NODE	ISN_REGION
REGION_CL	CITIZEN	ISN_NODE	ISN_REGION
REGION_CL	ADDRESS	ISN_NODE	ISN_REGION
REPLY	DOC_FOLDER_ITEM	ISN_REPLY	ISN_REPLY
REPLY	REMINDER	ISN_REPLY	ISN_REPLY
RESOLUTION	REPLY	ISN_RESOLUTION	ISN_RESOLUTION
RESOLUTION	DOC_FOLDER_ITEM	ISN_RESOLUTION	ISN_RESOLUTION
RESOLUTION	RESOLUTION_CARD	ISN_RESOLUTION	ISN_RESOLUTION
RESOLUTION	RESOLUTION_PRJ_COPY	ISN_RESOLUTION	ISN_RESOLUTION
RESOLUTION	RESOLUTION	ISN_RESOLUTION	ISN_PARENT_RESOLUTION
RESOLUTION_CATEGORY_CL	RESOLUTION	ISN_LCLASSIF	ISN_CATEGORY
RESPRJ_PRIORITY_CL	RESOLUTION	ISN_LCLASSIF	ISN_RESPRJ_PRIORITY
RESPRJ_STATUS_CL	RESOLUTION	ISN_LCLASSIF	RESPRJ_STATUS
RUBRIC_CL	REF_RUBRIC	DUE	DUE_RUBRIC
RUBRIC_CL	PRJ_REF_RUBRIC	DUE	DUE_RUBRIC
SECURITY_CL	USERSECUR	SECURLEVEL	SECURLEVEL
SECURITY_CL	PRJ_RC	SECURLEVEL	SECURLEVEL
SECURITY_CL	DOC_RC	SECURLEVEL	SECURLEVEL
SEND_PACKAGE	PACK_ITEMS	ISN_PACKAGE	ISN_PACKAGE
SEV_CHANNEL	SEV_PARTICIPANT	ISN_LCLASSIF	ISN_CHANNEL
SEV_PARTICIPANT	SEV_PARTICIPANT_RULE	ISN_LCLASSIF	ISN_PARTICIPANT
SEV_PARTICIPANT	SEV_SYNC_REPORT	ISN_LCLASSIF	ISN_PARTICIPANT

Главная таблица	Связанная таблица	Первичный ключ	Внешний ключ
SEV_REPORT_EVENT	SEV_REPORT	ISN_LCLASSIF	ISN_REPORT_EVENT
SEV_RULE	SEV_PARTICIPANT_RULE	ISN_LCLASSIF	ISN_RULE
SIGN_KIND_CL	REF_FILE_EDS	ISN_LCLASSIF	ISN_SIGN_KIND
SMEV_PARTICIPANT	SMEV_SETTINGS_REQUEST_VERSION	ISN_SMEV_PARTICIPANT	ISN_SMEV_PARTICIPANT
SMEV_SETTINGS_REQUEST	SMEV_SETTINGS_REQUEST_VERSION	ISN_SETTING_REQUEST	ISN_SETTINGS_REQUEST
SRCH_CRITERY	SRCH_CATEGORY	ISN_CRITERY	ISN_CRITERY
SRCH_GROUP	SRCH_GROUP_ITEM	ISN_GROUP	ISN_GROUP
SRCH_GROUP	USER_SRCH_GROUP	ISN_GROUP	ISN_GROUP
SRCH_REQUEST	SRCH_REQ_DESC	ISN_REQUEST	ISN_REQUEST
SRCH_REQUEST	USER_REQUEST	ISN_REQUEST	ISN_REQUEST
SRCH_REQUEST	EVNT_OBSERVED	ISN_REQUEST	ISN_REQUEST
SRCH_REQUEST	SRCH_GROUP_ITEM	ISN_REQUEST	ISN_REQUEST
SRCH_VIEW	SRCH_REQUEST	ISN_VIEW	ISN_VIEW
SRCH_VIEW	USER_VIEW	ISN_VIEW	ISN_VIEW
SRCH_VIEW	SRCH_VIEW_DESC	ISN_VIEW	ISN_VIEW
SSCAN_BARCODE	SSCAN_REQUEST	ISN_BARCODE	ISN_BARCODE
STATUS_EXEC_CL	RESOLUTION	ISN_LCLASSIF	ISN_STATUS_EXEC
STATUS_REPLY_CL	REPLY	ISN_LCLASSIF	ISN_STATUS_REPLY
STTEXT_LIST	STTEXT	ISN_STTEXT_LIST	ISN_STTEXT_LIST
STTEXT_LIST	STTEXT_CONTROL	ISN_STTEXT_LIST	ISN_STTEXT_LIST
UFOLDER	REF_UFOLDER	ISN_REF_UFOLDER	ISN_REF_UFOLDER
UFOLDER	UFOLDER_ACCESS	ISN_REF_UFOLDER	ISN_UFOLDER
USER_CL	USERSECUR	ISN_LCLASSIF	ISN_LCLASSIF
USER_CL	REF_SEND	ISN_LCLASSIF	ISN_SENDUSER
USER_CL	UFOLDER	ISN_LCLASSIF	ISN_USER

Главная таблица	Связанная таблица	Первичный ключ	Внешний ключ
USER_CL	USERCARD	ISN_LCLASSIF	ISN_LCLASSIF
USER_CL	USER_CABINET	ISN_LCLASSIF	ISN_LCLASSIF
USER_CL	USERDEP	ISN_LCLASSIF	ISN_LCLASSIF
USER_CL	USER_AUDIT	ISN_LCLASSIF	ISN_USER
USER_CL	USER_AUDIT	ISN_LCLASSIF	ISN_WHO
USER_CL	USER_REQUEST	ISN_LCLASSIF	ISN_USER
USER_CL	STTEXT_CONTROL	ISN_LCLASSIF	ISN_USER
USER_CL	USER_ORGANIZ	ISN_LCLASSIF	ISN_LCLASSIF
USER_CL	USER_DOCGROUP_ACCESS	ISN_LCLASSIF	ISN_LCLASSIF
USER_CL	NTFY_USER_EMAIL	ISN_LCLASSIF	ISN_USER
USER_CL	NTFY_EVENT	ISN_LCLASSIF	ISN_USER
USER_CL	USER_CERTIFICATE	ISN_LCLASSIF	ISN_USER
USER_CL	USER_VIEW	ISN_LCLASSIF	ISN_USER
USER_CL	USER_HISTORY	ISN_LCLASSIF	ISN_LCLASSIF
USER_CL	UFOLDER_ACCESS	ISN_LCLASSIF	ISN_GRANTEE
USER_CL	USER_TECH	ISN_LCLASSIF	ISN_LCLASSIF
USER_CL	USER_CERT_PROFILE	ISN_LCLASSIF	ISN_USER
USER_CL	USER_RIGHT_DOCGROUP	ISN_LCLASSIF	ISN_LCLASSIF
USER_CL	NEW_RECORD_CABINET	ISN_LCLASSIF	ISN_USER
USER_CL	USER_EDIT_ORG_TYPE	ISN_LCLASSIF	ISN_USER
USER_CL	USER_FILESECUR	ISN_LCLASSIF	ISN_LCLASSIF
USER_CL	USER_AUTH_EXTERNAL_ID	ISN_LCLASSIF	ISN_LCLASSIF
USER_CL	FORUM_READ	ISN_LCLASSIF	ISN_USER
USER_CL	BPM_PROCESS_VERSION	ISN_LCLASSIF	ISN_USER_LOCK
USER_CL	BPM_USER_PROCESS_TYPE	ISN_LCLASSIF	ISN_LCLASSIF

Главная таблица	Связанная таблица	Первичный ключ	Внешний ключ
USER_CL	MREVT_EVENT	ISN_LCLASSIF	INS_WHO
USER_CL	MREVT_EVENT	ISN_LCLASSIF	UPD_WHO
USER_CL	USER_SRCH_GROUP	ISN_LCLASSIF	ISN_USER
USER_LISTS	LIST_ITEMS	ISN_LIST	ISN_LIST
USERCARD	USER_CARD_DOCGROUP	ISN_LCLASSIF DUE	ISN_LCLASSIF DUE_CARD
VISA_TYPE_CL	PRJ_VISA_SIGN	ISN_LCLASSIF	ISN_VISA_TYPE
WAPI_SESSION	WAPI_SESSION_STORE	ISL_SID	ISL_SID
WF_INSTANCE	WF_SUBSCRIPTION	ISN_INSTANCE	ISN_INSTANCE
WF_INSTANCE	WF_INSTANCE_STATE	ISN_INSTANCE	ISN_INSTANCE
WF_LIBRARY	WF_PROCESS_TYPE	ISN_LIBRARY	ISN_LIBRARY
WF_PROCESS_TYPE	WF_PROCESS_CONFIG	ISN_PROCESS_TYPE	ISN_PROCESS_TYPE
WF_PROCESS_TYPE	WF_INSTANCE	ISN_PROCESS_TYPE	ISN_PROCESS_TYPE
WF_SERVICE	WF_PROCESS_CONFIG	ISN_SERVICE	ISN_SERVICE
WF_SERVICE	WF_INSTANCE	ISN_SERVICE	ISN_SERVICE
WF_SUBSCRIPTION	WF_SUBSCRIPTION_EVENT	ISN_INSTANCE SUBSCRIPTION_ID	ISN_INSTANCE SUBSCRIPTION_ID

Приложение Е

Специальные модификаторы данных

Е.1 Номерообразование

release_num_web – изменение счетчика номеробразования РК

Процедура освобождает зарезервированный номер в таблице номеробразования, который был занят при помощи reserve_num.

В таблице описаны аргументы хранимой процедуры (см. Таблица Е. 1).

Таблица Е. 1

Название	Тип	Описание	Допустимые значения
aduedocgroup	string(48)	Код due который подавали в reserve_num.	DOCGROUP_CL.DUE.
ayear	int	Значение года.	Целое число.
aordernum	smallint	Порядковый номер, полученный с помощью reserve_num.	Целое число.

При выполнении процедуры могут возникнуть следующие исключения:

– Доступ запрещен: пользователь (имя пользователя) не является пользователем системы «ДЕЛО».

release_prj_num_web – изменение счетчика номеробразования РКПД

Процедура освобождает зарезервированный номер в таблице номеробразования, который был занят при помощи reserve_prj_num.

В таблице описаны аргументы хранимой процедуры (см. Таблица Е. 2).

Таблица Е. 2

Название	Тип	Описание	Допустимые значения
aduedocgroup	string(48)	Код due который подавали в reserve_num.	DOCGROUP_CL.DUE.
ayear	int	Значение года.	Целое число.
aordernum	smallint	Порядковый номер, полученный с помощью reserve_num.	Целое число.

При выполнении процедуры могут возникнуть следующие исключения:

– Доступ запрещен: пользователь (имя пользователя) не является пользователем системы «ДЕЛО».

Е.2 Протокол

write_prot – протоколирование изменения и просмотра объектов Дела.

Процедура добавляет запись в протокол просмотра или изменения объекта в зависимости от идентификатора операции протоколирования.

В таблице описаны аргументы хранимой процедуры (см. Таблица Е. 3).

Таблица Е. 3

Название	Тип	Описание	Допустимые значения
aTable_Id	string(1)	Вида объекта протокола	
aOper_Id	string(1)	Вид операция протокола	
aSuboper_Id	string(1)	Подвид операции протокола	
aOper_Describe	string(3)	Уточнение операции протокола	
aRef_Isn	int	Идентификатор объекта	

Название	Тип	Описание	Допустимые значения
aOper_Comment	string(255)	Комментарий	

Аргументы aTable_Id, aOper_Id, aSuboper_Id, aOper_Describe вместе составляют уникальный ключ операции протоколирования.

При выполнении процедуры могут возникнуть следующие исключения:

- Доступ запрещен: пользователь (имя пользователя) не является пользователем системы «ДЕЛО».
- Неверный набор параметров: aTable_Id = (значение), aOper_Id = (значение), aSuboper_Id = (значение), aOper_Describe = (значение).

mark_read_prot – отметка прочтения пользователем объекта (РК, РКПД, поручения и т.д.).

Процедура (с версии 13.1) добавляет запись о факте просмотра текущим пользователем указанного объекта системы.

В таблице описаны аргументы хранимой процедуры (см. Таблица Е. 4).

Таблица Е. 4

Название	Тип	Описание	Допустимые значения
aTableID	string(1)	Вида объекта	D – РК документа R – поручение Y – отчет об исполнении поручения P – РКПД V – информация о визировании или подписании ПД F – прикрепленный файл
aRefISN	int	Идентификатор объекта	

При выполнении процедуры могут возникнуть следующие исключения:

- Доступ запрещен: пользователь (имя пользователя) не является пользователем системы «ДЕЛО».

Е.3 События

add_evnt_queue_item – добавление элементов в очередь событий

Процедура добавляет новое событие в очередь. Событие относится к объекту системы, ссылка на который указывается параметрами aObjectName и aObjectId. Значение параметра aObjectId формируется по следующим правилам:

- Если идентификатор объекта, это число (например, DOC_RC.ISN_DOC) – то в параметре aObjectId должна передаваться строка ISN#<ISN_DOC>.
- Если идентификатор это, код DUE – то в параметре aObjectId передается значение кода.

Если объект к которому относится сообщение это РК, РКПД или поручение, то в параметре aDueDocgroup должен передаваться код DUE соответствующей группы документов. Для остальных типов объектов этот параметр не заполняется.

В параметре aFlags может передаваться произвольная информация, но желательно передавать в нем список строковых идентификаторов, разделенных символом ",", (запятая): <flag1>,<flag2>,...,<flagN>

В таблице описаны аргументы хранимой процедуры (см. Таблица Е. 5).

Таблица Е. 5

Название	Тип	Описание	Допустимые значения
aKindEvent	int	Тип события	1 - создание, 2 - модификация
aObjectName	string(48)	Тип объекта	EOS_OBJECT.OBJECT_NAME
aObjectId	string(255)	Идентификатор объекта	Null, строка
aDueDocgroup	string(48)	Код due группы документов	Null, DOCGROUP_CL.DUE.
aFlags	string(2000)	Дополнительные флаги события	Список флагов

При выполнении процедуры могут возникнуть следующие исключения:

- Не задано значение параметра (Код due группы документов)
- Недопустимое значение параметра (Тип события)

save_custom_storage – сохранение данных в "произвольном" хранилище

Процедура изменяет данные в таблице CUSTOM_STORAGE, сохраняя переданное значение (вместо старого если оно существовало).

Для удаления из БД существующего значения нужно передать в качестве нового значения NULL.

Значение параметра aOwnerID формируется по следующим правилам:

- Если идентификатор объекта, это число (например, DOC_RC.ISN_DOC) – то в параметре aOwnerID должна передаваться строка ISN#<ISN_DOC>.
- Если идентификатор, это код DUE – то в параметре aOwnerID передается значение кода.

В таблице описаны аргументы хранимой процедуры (см. Таблица Е. 6).

Таблица Е. 6

Название	Тип	Описание	Допустимые значения
aValueID	string(255)	Идентификатор записи	Строка, уникально определяющая сохраняемое значение
aValue	string(2000)	Сохраняемое значение	Строка, Null – для удаления значений с указанным в параметре aValueID идентификатором из БД
aOwnerKind	int	Код объекта – владельца значения	EOS_OBJECT.ISN_EOS_OBJECTS, кроме таблицы RESOLUTION, для которой kind_object = 4
aOwnerID	string(255)	Идентификатор объекта – владельца значения	Null, строка

При выполнении процедуры могут возникнуть следующие исключения:

Доступ запрещен. Пользователь <ИМЯ> не является пользователем системы «ДЕЛО».

Приложение Ж

Нестандартные критерии поиска ДЕЛО API 2009

Для случаев нестандартных критериев поиска разработаны специальные SQL-запросы, которые могут вызываться пользователем.

Состав и функционал данных запросов фиксированный, определяется и поддерживается разработчиками системы «ДЕЛО».

Список параметров запросов, которые реализованы как SQL-код в ДЕЛО API 2009 (см. Таблица Ж. 1):

Таблица Ж.1

Параметр	Реквизит	Тип ¹⁾	Условия ²⁾	Примечание
Rc.RegNum	Rc.RegNum + Rc.OrderNum	Строка; Число	Many; Range	Регистрационный или порядковый номер документа (входящего и/или исходящего). Если значение – строка, то условие задается только на Rc.RegNum, если число, то это значение ищется как среди регистрационных номеров, так и среди порядковых номеров документа. Дополнительным разделителем поисковых образов в этом параметре является пробел.
Rc.DocContents		Слова		Контекстный поиск текста в прикрепленных к регистрационной карточке (РК) файлах (без учета файлов отчетов исполнителей поручений).
Rc.UserRead		Число	Many	Системный номер пользователя, который имеет в Журнале отметку о прочтении РК. Если задан набор системных номеров, тогда у РК должна быть отметка о прочтении хотя бы одним из этих пользователей.
Rc.UserUnRead		Число	Many	Системный номер пользователя, который не имеет в Журнале отметку о прочтении РК. Если задан набор системных номеров, тогда у РК нет отметки о прочтении ни одним из этих пользователей.
Rc.UserCanRead (DOC_RC.UserCanR ead)		Число		Системный номер пользователя, который имеет доступ к РК на чтение.
RcPrj.RegNum	RcPrj.RegNum +	Строка;	Many;	Регистрационный или

Параметр	Реквизит	Тип ¹⁾	Условия ²⁾	Примечание
	RcPrj. OrderNum	Число	Range	порядковый номер проекта документа. Если значение – строка, то условие задается только на RcPrj.RegNum, если число, то условие задается на RcPrj.OrderNum. Дополнительным разделителем поисковых образов в этом параметре является пробел.
RcPrj.UserRead (PRJ_RC.UserRead)		Число	Many	Системный номер пользователя, который имеет в Журнале отметку о прочтении РКПД. Если задан набор системных номеров, тогда у РКПД должна быть отметка о прочтении хотя бы одним из этих пользователей.
RcPrj.UserUnRead (PRJ_RC.UserRead)		Число	Many	Системный номер пользователя, который не имеет в Журнале отметку о прочтении РКПД. Если задан набор системных номеров, тогда у РКПД нет отметки о прочтении ни одним этих пользователей.
Rcprj.UserCanRead (PRJ_RC.UserCanRead)		Число		Системный номер пользователя, который имеет доступ к РКПД на чтение.
Cor.Region	Correspondent. Organiz.Region.DC ode (DUE)	Строка	Many; Begins; Null	Код региона корреспондента-организации или автора сопроводительного документа.
Cor.City	Correspondent. Organiz.City	Текст		Город организации – корреспондента документа.
Cor.Address	Correspondent. Organiz. Address	Текст		Адрес организации – корреспондента документа.
Cor.Name	Correspondent. Organiz.Name	Текст		Наименование организации – корреспондента документа.
Cor.OutNum (Cor.OutNumExact)	Correspondent. OutNum	Текст		Исходящий номер корреспондента или автора сопроводительного документа.
Journal.RecordExist (DOC_RC/Journal.Re cordExist)		True; False		Флаг наличия записи в Журнале передачи документа (ЖПД): True – запись в Журнале существует; False – записи в Журнале нет. (При наличии данного критерия в поисковом запросе

Параметр	Реквизит	Тип ¹⁾	Условия ²⁾	Примечание
				– другие критерии на записи Журнала игнорируются).
Journal.KindAbsent	Journal. Kind	Число	Many	Отсутствие записи в ЖПД определенного вида: 1 – кандидат для передачи документа; 2 – запись о передаче документа; 3 – запись о списании документа в АИК «НАДЗОР-WEB»; 4 – запись об уничтожении документа. (При наличии данного критерия в поисковом запросе – другие критерии игнорируются).
Resolution.AuthorOuter		Число	Many	Системный номер организации или контакта – внешнего Автора резолюции.
Resolution. UserRead		Число	Many	Системный номер пользователя, который имеет в Журнале отметку о прочтении Поручения. Если задан набор системных номеров, тогда у Поручения должна быть отметка о прочтении хотя бы одним из этих пользователей.
Resolution. UserUnRead		Число	Many	Системный номер пользователя, который не имеет в Журнале отметку о прочтении Поручения. Если задан набор системных номеров, тогда у Поручения нет отметки о прочтении ни одним из этих пользователей.
Resolution.UserCanRead (RESOLUTION.UserCanRead)		Число		Системный номер пользователя, который имеет доступ к поручению на чтение.
Reply.PersonOuter (DOC_RC/Reply.PersonOuter)		Число	Many	Системный номер организации или представителя – внешнего Исполнителя поручения.
Reply. UserRead		Число	Many	Системный номер пользователя, который имеет в Журнале отметку о прочтении Отчета об исполнении поручения. Если задан набор системных номеров, тогда у Отчета об исполнении поручения должна быть отметка о

Параметр	Реквизит	Тип ¹⁾	Условия ²⁾	Примечание
				прочтении хотя бы одного из этих пользователей.
Reply.UserUnRead		Число	Many	Системный номер пользователя, который не имеет в Журнале отметку о прочтении Отчета об исполнении поручения. Если задан набор системных номеров, тогда у Отчета об исполнении поручения нет отметки о прочтении ни одним из этих пользователей.
DocKind ³⁾		In; Let; Both; Out		Признак того, какие документы должны отбираться (имеет смысл только при поиске документов): In – только входящие документы (без писем); Let – только входящие письма; Both – ищутся все входящие документы (вместе с письмами); Out – Ищутся только исходящие документы.

¹⁾ – **Тип передаваемого значения** – строка, но значение этой строки должно без ошибок конвертироваться в указанный тип. Например, если указан тип "Дата", то значение строки должно быть в формате "ДД/ММ/ГГГГ", "Число" - строка должна конвертироваться в целое число (чаще всего в неотрицательное), "Строка" - длина одного задаваемого значения не должна превышать 250 байт.

Тип "Слова" - значение задаваемой строки разбивается системой на несколько значений "слова" по символам-разделителям: "пробел", "кавычки", а также любой из следующих символов "!&()+,./:;'?|=\\". Условие поиска каждого слова – **вхождение** его в текст данного реквизита.

Тип "Текст" - определяет новый синтаксис значения этих параметров, который аналогичен синтаксису, используемому в задаче "Поиск".

²⁾ – **Many** – допустимо задавать несколько значений (с условиями) на один реквизит, соединенных условием "ИЛИ" и перечисленных через "|". Например: "1|3|5" или "01/01/1997:01/12/1997|31/12/1997".

Begins – допустима строка с "%" в конце – условие на то, что значение данного реквизита должно начинаться с указанной строки. Обычно это условие используется для поиска информации по агрегированному значению классифицированного реквизита (например, если в справочнике «Организации» есть вершина "Федеральные органы власти" под которой есть конкретные организации, то можно найти все документы, корреспондентом которых являлись Федеральные органы власти, хотя корреспондентами документа являются конкретные организации (Params("Cor.Organiz") = Dcode + "%")).

Range – допустимо задание двух значений, соединенных знаком ":" – условие на то, что значение реквизита должно принадлежать указанному диапазону значений (включая границы) Если необходимо задать открытый диапазон, значение

соответственной границы должно иметь значение Null, например: "Null:0" означает, что нужно найти все значения реквизита, имеющие не положительные значения.

Null - допустимы значения "IsNull" и "IsNotNull" (символы в любом регистре) – условие на заполненность или незаполненность соответствующего реквизита.

³⁾ – **Признак DocKind** – необязательная опция для поиска документов. Если она не определена, то ищутся все документы, удовлетворяющие заданным условиям. Но в некоторых случаях, когда необходимо отобрать документы только определенного вида, задание этой опции может значительно уменьшить время поиска.

* – Звездочкой отмечены реквизиты, условия на которые рекомендуется задавать как дополнительные к основным, как правило это реквизиты типа "строка".

Приложение И

Синтаксис запросов на текстовые поля ДЕЛЮ API 2009

Поисковый запрос – это то, что пользователь ввел в поле поисковой формы с удаленными пробелами слева и справа и дополненный одним пробелом слева и справа:

(" " !! trim (поле) !! " ")

<поисковый запрос> ::= <поисковый образ> | <поисковый запрос> <оператор>
<поисковый образ>

<поисковый образ> ::= <ПРОБЕЛ><ПО><ПРОБЕЛ> |
<ПРОБЕЛ><НОНЕ><ПО><ПРОБЕЛ>

<ПО> ::= <строка> | <строка в кавычках> | <оператор позиции><строка> |
<оператор позиции><строка в кавычках> | <ПУСТО>

<оператор> ::= <И><ИЛИ>

<оператор позиции> ::= <начало поля> | <конец поля> | <точно равно>

<строка> ::= <допустимый символ> | <строка><допустимый символ>

<строка в кавычках> ::= "<строка>"

<допустимый символ> ::= а | б | | /* любой символ, отличный от пробела, допустимый в фразе LIKE, кавычки в строке (если имеются) изображаются, как двойные "" */

<И> ::= &

<ИЛИ> ::= !

<НОНЕ> ::= ^

<начало поля> ::= <

<конец поля> ::= >

<точно равно> ::= =

<ПУСТО> ::= -

<ПРОБЕЛ> ::= "" | <ПРОБЕЛ>" " /* пробел не обязателен до и после разделителя, несколько пробелов эквивалентны одному пробелу */

Приложение К

XSD-схемы файлов, используемых при организации электронного взаимодействия с системой «ДЕЛО»

В системе «ДЕЛО» используется два стандартных формата электронных документов, предназначенных для организации взаимодействия между организациями:

- для отправки/получения сведений о документах по электронной почте
- для обеспечения обмена с помощью подсистемы СЭВ

К.1 Формат обмена по электронной почте.

XSD – схемы документов, используемых для обмена по электронной почте, введены в состав ПО системы «ДЕЛО» только начиная с версии 12.0. Поэтому, хотя стандарты языка XML рекомендуют объявлять внутри документа используемое пространство имен (в нашем случае <http://www.eos.ru/2011/rc-email>), этого не следует делать для паспортов, предназначенных для регистрации в системе «ДЕЛО» версии младше 12.0, так как это может привести к ошибкам парсера XML и некорректному заполнению реквизитов РК, регистрируемой из паспорта. Однако для этих версий данные XSD - схемы также можно использовать для промежуточной валидации формируемых паспортов.

XSD-схемы документов, а также их описания включены в состав дистрибутива системы. Их можно найти в каталоге \SDK\XSD\Email:

- Passport.xsd - XSD-схема паспорта.
- Receipt.xsd - XSD-схема уведомления о регистрации.
- Refusal.xsd - XSD-схема уведомления об отказе от регистрации.
- Common.xsd - Общие элементы XSD-схем документов.
- PassportPrj.xsd – XSD-схема паспорта. Данный формат используется при

выгрузке РКПД.

Также в наборе документации присутствует файл «Описание формата сообщения E-mail.doc».

К.2 Формат обмена подсистемы СЭВ

Подсистема СЭВ использует в своей работе сообщения четырех видов: паспорт, доклад, сведения для синхронизации справочных данных, доклад о получении данных синхронизации справочников. При отправке и получении сообщений, подсистема выполняет их валидацию по XSD-схемам. В случае ошибки валидации некорректные сообщения сохраняются в архиве СЭВ, но их дальнейшая обработка не выполняется. Т.е. исходящие сообщения не отправляются, а входящие не регистрируются.

XSD-схемы сообщений подсистемы СЭВ, а также их описания включены в состав дистрибутива системы. Их можно найти в каталоге \SDK\XSD\Sev:

- DocumentInfo.xsd – схема паспорта документа.
- ReportInfo.xsd – схема доклада СЭВ.
- SyncInfo.xsd – схема сведений для синхронизации справочных данных.
- SyncReport.xsd – схема доклада о получении данных синхронизации

справочников.

- CommonInfo.xsd – общие элементы XSD-схем подсистемы СЭВ.
- Envelop.xsd – схема конверта, с которым передается сообщение СЭВ и

который является обязательным элементом комплекта сообщения. Конверт формируется в файле формата XML с расширением «.env».

– SevExt2015-2020.xsd – расширения схемы паспорта документа при добавлении новых полей.

– RemoveOtherNamespacesV3.xslt – файл преобразования для очистки XML-документа от спец. пространств имен, с которыми добавлялись новые поля.

Также в наборе документации присутствует файл «Описание формата сообщения СЭВ.doc»

Приложение Л

Описание механизма событий БД «ДЕЛО»

Задачи, связанные с обработкой документов, часто требуют реализации функций отслеживания появления объектов в системе и изменения состояния объектов. Под понятием "объект" может иметься в виду: РК документа, РК проекта документа, поручение, элемент какого-либо справочника.

Л.1 Возможности, предоставляемые механизмом событий

- Доставка событий для фоновой задачи происходит в асинхронном режиме. Т.е. от момента возникновения события до момента его обработки может пройти неопределенный интервал времени.
- Поддержка событий создания и модификации объектов (удаление объекта рассматривается как модификация)
- Отслеживается состояние следующих объектов:
 - 1) РК документа
 - 2) РК проекта документа
 - 3) Поручение
 - 4) Элемент справочника

Л.2 Описание алгоритмов регистрации событий в очередях

Реализация основана на использовании механизма триггеров СУБД. При этом используется три различных алгоритма регистрации событий, в зависимости от классификации объектов по структуре данных и по способу работы приложений системы «ДЕЛО» с объектами:

- **Простой объект.** Объект, данные которого сохраняются в единственной таблице базы данных. Каждая запись этой таблицы, таким образом, представляет собой экземпляр объекта.
- **Сложный объект.** Объект, данные которого сохраняются в нескольких таблицах, связанных между собой ссылками. При этом можно выделить таблицу, записи которой представляют "корень объекта", и дочерние таблицы – содержащие ссылку на запись корневой таблицы объекта, либо на запись другой дочерней таблицы этого же объекта.
- **Сложный объект с событием завершения сохранения.** Структура хранения аналогична сложному объекту. Но, в отличие от сложных объектов, после создания или модификации данных объекта, приложение вызывает специальную хранимую процедуру, позволяющую выполнить необходимые действия с объектом на уровне БД. Хранимая процедура НЕ ВЫЗЫВАЕТСЯ при удалении таких объектов, т.к. при удалении можно пользоваться возможностями триггеров БД, срабатывающих на событие удаления записи из корневой таблицы объекта.

Регистрация событий простых объектов

На каждую добавляемую, модифицируемую или удаляемую строку производится регистрация события со следующими данными:

- Если тип события – добавление записи, то в процедуру в параметре "Вид события" передается значение – 1, при модификации или удалении – 2.
- В параметре "Вид объекта" передается код, взятый из таблицы EOS_OBJECT
- В параметре "Идентификатор объекта" передается значение первичного ключа соответствующей записи таблицы
- В параметре "Группа документов" передается значение NULL

– В параметре "Флаги операции" передается значение стандартного флага (см. описание стандартных флагов ниже)

Регистрация событий сложных объектов

Алгоритм реализуется в триггерах, срабатывающих на добавление, изменение и удаление записей в таблицах, хранящих данные объекта: как на корневой таблице, так и на дочерних таблицах.

Триггеры, подключенные к корневой таблице, работают аналогично триггерам регистрации событий простых объектов.

Триггеры, подключенные к дочерней таблице, также вызывают хранимую процедуру регистрации событий один раз на каждую затронутую строку. Используются следующие значения аргументов:

- В параметре "Вид события" всегда передается 2.
- В параметре "Вид объекта" передается код соответствующий "корневой" таблице объекта, взятый из таблицы EOS_OBJECT, кроме таблицы RESOLUTION, для которой kind_object = 4
- В параметре "Идентификатор объекта" передается значение первичного ключа записи "корневой" таблицы (см. описание поля "Идентификатор объекта" таблицы хранения данных очередей событий).
- В параметре "Группа документов" передается значение NULL
- В параметре "Флаги операции" передается значение стандартного флага (см. описание стандартных флагов ниже)

Никаких методов, предназначенных для уменьшения количества записей в таблице событий, не применяется. Например, если пользователь через приложение выполняет создание экземпляра сложного объекта, с одновременным заполнением каких-либо данных в дочерних таблицах данного объекта, будет происходить следующее:

- Пользователь редактирует форму создания объекта.
- Пользователь выполняет действие "Сохранить" для объекта.
- Приложение выполняет операцию INSERT в "корневую" таблицу объекта.
- Приложение выполняет операцию INSERT в "дочерние" таблицы объекта, в зависимости от введенных пользователем данных, запись может быть выполнена в одну или несколько таблиц.

При этом в таблице регистрации событий будет созданы следующие записи:

- Запись со значением 1 в поле "Вид события" созданная из триггера на добавление записи в корневую таблицу объекта.
- Одна или больше записей со значением 2 в поле "Вид события", созданные из триггеров на добавление.

Регистрация событий сложных объектов с событием завершения сохранения

Таковыми объектами в системе являются:

- РК документа – в конце сохранения данных вызывается процедура RC_SAVE
- РК проекта документа – в конце сохранения данных вызывается процедура PRJ_SAVE
- Поручение – в конце сохранения данных вызывается процедура RES_SAVE
- Пользователь – в конце сохранения данных вызывается процедура USER_SAVE

Для таких объектов модифицированы соответствующие процедуры сохранения, в них добавлен код вызова процедуры регистрации событий.

Например, если пользователь через приложение выполняет создание экземпляра такого объекта, с одновременным заполнением каких-либо данных в дочерних таблицах данного объекта, будет происходить следующее:

- Пользователь редактирует форму создания объекта.
- Пользователь выполняет действие "Сохранить" для объекта.
- Приложение выполняет операцию INSERT в "корневую" таблицу объекта.
- Приложение выполняет операцию INSERT в "дочерние" таблицы объекта, в зависимости от введенных пользователем данных, запись может быть выполнена в одну или несколько таблиц.

- Приложение выполняет вызов соответствующей хранимой процедуры.

При выполнении хранимой процедуры будет вызвана процедура регистрации события со значением 1 в параметре "Вид события".

При модификации объекта пользователем, выполняемые действия – следующие:

- Пользователь редактирует форму редактирования объекта.
- Пользователь выполняет действие "Сохранить" для объекта.
- Приложение выполняет операцию UPDATE для "корневой" таблицы объекта (при необходимости).

– Приложение выполняет операции INSERT, UPDATE, DELETE для "дочерних" таблиц объекта (при необходимости).

- Приложение выполняет вызов соответствующей хранимой процедуры.

При выполнении хранимой процедуры будет вызвана процедура регистрации события со значением 2 в параметре "Вид события".

При удалении объектов, хранимые процедуры не вызываются, поэтому, вызов процедуры регистрации события делается в триггере, срабатывающем на удаление записей из "корневой" таблицы объекта.

Вызов процедуры регистрации событий выполняется со следующими значениями аргументов:

– Если тип события – добавление записи, то в процедуру в параметре "Вид события" передается значение – 1, иначе – 2.

– В параметре "Вид объекта" передается код, взятый из таблицы EOS_OBJECT, кроме таблицы RESOLUTION, для которой kind_object = 4

– В параметре "Идентификатор объекта" передается значение первичного ключа соответствующей записи таблицы (см. описание поля "Идентификатор объекта" таблицы хранения данных очередей событий)

– В параметре "Группа документов" передается значение DUE_DOCGROUP соответствующего объекта. Для поручений используется группа документов РК, которой поручение принадлежит.

– В параметре "Флаги операции" передается список флагов, установленных в триггерах при модификации таблиц, входящих в данный составной объект, и стандартные флаги операций.

Стандартные флаги редактирования сущностей.

Имена флагов формируются по схеме:

<I|U|D>_<TABLE_NAME>[_<OBJ>], где:

– I, U, D – признак операции – добавление, изменение, удаление соответственно.

– TABLE_NAME – имя таблицы БД – например, DOC_RC, REF_SEND, REPLY и т.п.

– OBJ – только для таблицы REF_FILE – суффикс определяющий, к какому объекту принадлежит запись. Возможные значения: RC, RES, PRJ.

Примеры флагов:

- I_REF_CORRESP – была добавлена запись в раздел Корреспонденты РК.

- D_FORWARD – была выполнена операция отмены пересылки.
- I_REF_FILE_RES – добавлен файл отчета исполнителя резолюции.

Также, в флагах отмечается факт изменения дополнительных реквизитов РК и РКПД – в поле FLAGS пишется API_NAME измененного реквизита (см. Таблица Л. 1).

Таблица Л. 1 - Перечень флагов операций в событиях сложных объектов

Флаг	Описание
marksend	Проставляется, когда ожидается отправка для внешнего или внутреннего адресата (дата отправки заполнена).
forward	Проставляется при добавлении записи в журнале пересылок РК.
insrc	Проставляется при создании РК.
rcdel	Проставляется при удалении РК.
rcctrl	Проставляется при возникновении события, могущего повлиять на состояние контрольности РК.
adjlayer	Проставляется при переносе резолюции или куста резолюций на другой уровень.
outcard	Проставляется при добавлении или изменении исполнителя документа.
carddel	Проставляется при возникновении события, могущего привести к исключению РК из картотеки.
calcsend	Проставляется при добавлении, изменении или удалении адресата РК.
prjnewstage	Проставляется при создании новой версии РКПД.
resadd	Проставляется при добавлении резолюции, проекта резолюции и рассылке проекта резолюции.
rescard	Проставляется при изменении принадлежности поручения картотекам.
resdel	Проставляется при удалении поручения.
resprjdel	Проставляется при удалении проекта резолюции.
resctrlend	Проставляется при снятии поручения с контроля.
prjrcdel	Проставляется при удалении РКПД.
to_reestr	Включение РК в реестр или исключение РК из реестра
send_reestr	Реестр получил статус "отправлен"
mark_email	В РК отмечена "без реестровая" отправка, например, отправка по email
ACCEPT_EXECUTE	Поручение принято исполнителем на исполнение – заполнено поле REPLY.RECEPT_DATE
res_prj_approved_or_sent	Проект поручения был утвержден руководителем или утвержден и разослан

Приложение М

Список имен объектов и их идентификаторов (KIND_OBJECT)

Список имен объектов и их идентификаторов (KIND_OBJECT) представлен в таблице ниже (см. Таблица М.1).

Таблица М. 1

Имя объекта	Код объекта (KIND_OBJECT)
ACQUAINTANCE	131
ADDR_CATEGORY_CL	349
ADDRESS	742
APP_HOST	740
APPROACH	518
AR_CATEGORY	611
AR_CLS	654
AR_CLS_CONTROL	655
AR_CLS_FIELD	656
AR_DESCRIPT	513
AR_DOCGROUP	514
AR_ORGANIZ_VALUE	733
AR_PRJ_VALUE	657
AR_RC_VALUE	515
AR_RUBRIC_VALUE	516
AR_VALUE_LIST	517
ARCH_CLASSIFY_CL	578
ARCH_DEPARTMENT_CL	558
ARCH_DESTRUCT_ACT	589
ARCH_DOCUMENT	567
ARCH_DOCUMENT_OD	581
ARCH_DOCUMENT_RUBRIC	591
ARCH_DOCUMENT_RUBRIC_OD	593
ARCH_FUND_CL	572
ARCH_INVENTORY_KIND_CL	563
ARCH_JOURNAL	562
ARCH_KEEPING_CL	579
ARCH_KEEPINGTYPE_CL	580
ARCH_LIST_CL	560
ARCH_NOM_EXT_RUBRIC	590
ARCH_NOM_EXT_RUBRIC_OD	592
ARCH_NOMENKL	561
ARCH_NOMENKL_EXT	568
ARCH_NOMENKL_EXT_OD	8

Имя объекта	Код объекта (KIND_OBJECT)
ARCH_NOMENKL_LIST	602
ARCH_NOMENKL_LIST_OD	603
ARCH_NOMENKL_PECULIARITY	569
ARCH_NOMENKL_STATUS	576
ARCH_ORGANIZATION_CL	577
ARCH_PECULIARITY_CL	570
ARCH_REF_FILE	594
ARCH_REF_FILE_OD	586
ARCH_RETIREMENT_CL	573
ARCH_STORE_CL	601
ARCH_SUM_INVENTORY_CL	566
BANK_RECVISIT	658
BAR_CODE_SUPPORT	499
BUF_FOLDER	704
BUF_MESSAGE	702
BUF_MESSAGE_FIELDS	703
BUF_RULE	706
BUF_USER_FOLDER	705
BULK_DELETE_PROT	638
CA_CATEGORY	720
CABINET	120
CALENDAR_CL	643
CB_PRINT_INFO	713
CERTIFICATE	582
CHECKIN	659
CITIZEN	109
CITIZEN_STATUS	488
CITSTATUS_CL	487
CL_SEARCH	737
CLASSIF_LIST	519
CONTACT	630
CRITERION	660
DELIVERY_CL	112
DELO_BLOB	715
DELO_OWNER	520
DEPARTMENT	104
DEPARTMENT_REPL	738
DG_FILE_CONSTRAINT	700

Имя объекта	Код объекта (KIND_OBJECT)
DIC_BUF	661
DIC_OBJ_BUF	663
DIC_OBJECT	662
DOC_DEFAULT	634
DOC_DEFAULT_VALUE	635
DOC_EXE	689
DOC_FOLDER_ITEM	525
DOC_RC	1
DOC_SIGN	618
DOC_TEMPLATES	521
DOC_WHO	617
DOCGROUP_CL	105
EDS_CATEGORY_CL	722
EOS_DB_REGISTRATION	664
EOS_OBJECT	665
EOS_TASKS	522
EVNT_FEED	712
EVNT_QUEUE_ITEM	506
EVNT_SUBSCRIPTION	505
EXT_PROT	651
EXT_PROT_OPER_TYPES	652
EXT_PROT_SETTINGS	653
FIG_ROLE_CL	610
FIGURANT	609
FILE_CONTENTS	523
FILE_CONTENTS2	666
FOLDER	524
FORMAT_CL	668
FORUM_MSG	698
FORUM_THEME	697
FORWARD	352
JOURNAL	121
LINK_CL	123
LIST_ITEMS	526
MESSAGES	527
MGF_EXCH_MESSAGE	11
MODULES	529
NEW_RECORD_CABINET	741

Имя объекта	Код объекта (KIND_OBJECT)
NOMENKL_CL	119
NOTIFY_CL	588
NTFY_EVENT	644
NTFY_MESSAGE	646
NTFY_OPERATION	642
NTFY_PERIODICITY	641
NTFY_SUBSCRIPTION	647
NTFY_SYSTEM_PARAMS	649
NTFY_USER_EMAIL	648
NUMCREATION	530
OBJ_TEMPLATE	699
OBJECT_USE	531
ORG_TYPE_CL	486
ORGANIZ_CL	106
PACK_ITEMS	354
PATCH_HISTORY	532
PLUGINS	670
PRJ_CURR_PRJ	600
PRJ_DEFAULT	636
PRJ_DEFAULT_VALUE	637
PRJ_EXEC	640
PRJ_FOLDER_ITEM	597
PRJ_FORUM	500
PRJ_NUMCREATION	599
PRJ_RC	7
PRJ_REF_RUBRIC	501
PRJ_REF_SEND	596
PRJ_RETURN	734
PRJ_VISA_SIGN	595
PROGRAMS	533
PROT	534
PROT_FILE_SSCAN	671
PROT_NAME	535
PROT_STREAM_SCAN	672
RECEIPT_COUNT	744
REESTR_NEW	481
REESTRTYPE_CL	124
REF_ACCESS_CARD	118

Имя объекта	Код объекта (KIND_OBJECT)
REF_CONTEXT	694
REF_CORRESP	108
REF_FILE	128
REF_FILE_ACCESS	639
REF_FILE_EDS	583
REF_LETTER	110
REF_LINK	129
REF_RUBRIC	548
REF_SEND	536
REF_SOISP	115
REF_UFOLDER	537
REF_VISA	116
REGION_CL	333
REMINDER	625
REP_NUMCREATION	673
REP_NUMCREATION2	674
REPEAT_RES	714
REPLY	117
REQUEST	675
RESOLUTION	4
RESOLUTION_CARD	130
RESOLUTION_CATEGORY_CL	616
RESPRJ_PRIORITY_CL	710
RESPRJ_STATUS_CL	711
RUBRIC_CL	107
SECURITY_CL	111
SEND_PACKAGE	353
SEV_ASSOCIATION	682
SEV_CHANNEL	686
SEV_COLLISION	695
SEV_PARTICIPANT	683
SEV_PARTICIPANT_RULE	684
SEV_REPORT	688
SEV_REPORT_EVENT	687
SEV_RULE	685
SEV_SUBSCRIBE	739
SEV_SYNC_REPORT	709
SHABLON_CL	484

Имя объекта	Код объекта (KIND_OBJECT)
SHABLON_DETAIL	550
SIGN_KIND_CL	622
SOFTWARE	676
SQLCOMMANDS	538
SRCH_AR_HIER	621
SRCH_CATEGORY	613
SRCH_CRITERY	612
SRCH_REQ_DESC	615
SRCH_REQUEST	614
SRCH_VIEW	691
SRCH_VIEW_DESC	693
STATUS_EXEC_CL	623
STATUS_REPLY_CL	624
STOP_WORDS	539
STTEXT	627
STTEXT_CONTROL	628
STTEXT_LIST	629
SW_LIST	677
SW_MODULES	678
SW_USE	679
T_WORDS	540
T_WU	541
TEMP_RC	701
UFOLDER	542
UFOLDER_ACCESS	707
USER_AUDIT	587
USER_CABINET	544
USER_CARD_DOCGROUP	543
USER_CERT_PROFILE	721
USER_CL	125
USER_DOCGROUP_ACCESS	633
USER_EDIT_ORG_TYPE	743
USER_HISTORY	696
USER_LISTS	545
USER_ORGANIZ	632
USER_PARDS	546
USER_REQUEST	626
USER_RIGHT_DOCGROUP	726

Имя объекта	Код объекта (KIND_OBJECT)
USER_TECH	708
USER_VIEW	692
USERCARD	126
USERDEP	631
USERSECUR	402
V_DEP_CABINET	549
VIEWPROT	650
VISA_TYPE_CL	598
WAPI_SESSION	680
WEIGHT_DUE	681
WF_INSTANCE	512
WF_INSTANCE_STATE	504
WF_LIBRARY	507
WF_PROCESS_CONFIG	509
WF_PROCESS_TYPE	508
WF_SERVICE	510
WF_SUBSCRIPTION	511

Приложение Н

Права пользователя

Н.1 Статический класс расширения UserExtensions

Список публичных методов (см. Таблица Н. 1):

Таблица Н. 1

Наименование	Возвращает	Параметры	Описание
GetDeloRightFlag	char	AbsoluteRight right	Получение значения флага из абсолютных прав системы «ДЕЛО»
HasDeloRight	bool	AbsoluteRight right	Проверка доступности абсолютного права системы «ДЕЛО»
GetTechRightFlag	char	TechRight right	Получение значения флага из прав системного технолога системы «ДЕЛО»
HasTechRight	bool	TechRight right	Проверка доступности права системного технолога системы «ДЕЛО»
HasBpmProcManageRight	bool	long? isnProcessType	Проверка наличия права управление бизнес-процессами
HasDepartmentsRight	bool	AbsoluteRight right	Возвращает true, если хотя бы одно из ДЛ пользователя имеет указанное право
CanTech	IDictionary<IWapiSession, IWapiSessionAcl>	IEnumerable<IWapiSession> sources, MiddlewareContextBase baseContext	Список прав доступа

Н.2 Класс UserRightsHelperService

Данный класс получаем через DataContextHelperService, метод DataContextHelper.UserRightsHelper()

Список публичных методов (см. Таблица Н. 2):

Таблица Н. 2

Наименование	Возвращает	Параметры	Описание
--------------	------------	-----------	----------

Наименование	Возвращает	Параметры	Описание
HasTechRightForAsync	bool	TechRight right, string due	Проверка доступности права системного технолога системы «ДЕЛО»
HasDeloRightForAsync	bool	AbsoluteRight right, string dueDep	Проверка доступности абсолютного права системы «ДЕЛО» (функции) по отношению к должностному лицу или организации
CanAddResolutionAsync	bool	string dueDep	Может добавлять резолюции от имени должностного лица или организации
CanAddResolutionProjectAsync	bool	string dueDep	Может добавлять проекты резолюции от имени должностного лица или организации
CanExecResolutionAsync	bool	string dueDep	Может исполнять резолюции от имени должностного лица или организации
CanControlResolutionAsync	bool	string dueDep	Может контролировать исполнение поручений от имени должностного лица или организации
CanViewConfidentialRcAsync	bool	string dueDep	Может читать конфиденциальные РК за должностное лицо
HasAddEdsRightAsync	bool		Право добавлять Электронную подпись.

* **примечание:** последний параметр всех async методов - cancellationToken.

Н.3 Класс PrjRcSecurityService

Данный класс получаем через DataContextHelperService, метод DataContextHelper.PrjRcSecurity(PrjRc)

Список публичных методов (см. Таблица Н. 3):

Таблица Н. 3

Наименование	Возвращает	Параметры	Описание
CheckPrjRcSecurlevelAsync	bool	long securlevel	Проверяет securlevel для текущего пользователя и наличие флага PrjRcSecurlevelConsider.
CanExecProjectAsync	bool		У текущего пользователя есть право Полное Чтение за ДЛ, который является Исполнителем проекта

Наименование	Возвращает	Параметры	Описание
CanWorkWithPrjAsync	bool		У текущего пользователя есть право Полное Чтение за ДЛ с правом работы с проектом документа
CanManageExecAsync	bool		У текущего пользователя есть право Полное Чтение за ДЛ с правом управления исполнителями проекта документа
CanManageEndorseAsync	bool		У текущего пользователя есть право Полное Чтение за ДЛ с правом управления визирующими проекта документа
CanManageSignAsync	bool		У текущего пользователя есть право Полное Чтение за ДЛ с правом управления подписывающими проекта документа
CanWorkWithFilesAsync	bool		У текущего пользователя есть право Полное Чтение за ДЛ с правом работы с файлами проекта документа
CanWorkWithFilesAndManageEndorseAsync	bool		У текущего пользователя есть право Полное Чтение за ДЛ с правом работы с файлами и управления визирующими проекта документа
CanWorkWithFilesAndManageSignAsync	bool		У текущего пользователя есть право Полное Чтение за ДЛ с правом работы с файлами и управления подписывающими проекта документа
CanWorkForAuthorAsync	bool		Текущий пользователь является автором проекта документа
IsAllowRefFileChangeVisaAsync	bool		Разрешено изменение файлов проекта документа среди визирующих
IsAllowRefFileChangeSignAsync	bool		Разрешено изменение файлов проекта документа среди подписывающих
CanReadAsync	bool		Есть право чтения
CanRegisterNotFigurantAsync	bool		Можно регистрировать документ не будучи фигурантом.

* **примечание:** последний параметр всех async методов - cancellationToken.

Н.4 Класс DocRcSecurityService

Данный класс получаем через DataContextHelperService, метод DataContextHelper.DocRcSecurity(DocRc).

Список публичных методов (см. Таблица Н. 4):

Таблица Н. 4

Наименование	Возвращает	Параметры	Описание
RegisteredTodayAsync	bool		Зарегистрировано сегодня
HasDocgroupSecurAccess	bool		Проверка по Securlevel и группе документов
CanEditAsync	bool		Редактирование РК
CanReadAsync	bool		Проверка наличия прав на чтение РК
CanRegister	bool		Есть право регистрации
RcInCurrentCardAsync	bool		РК в текущей картотеке
RcInRegAsync	bool		
CanDeleteAsync	bool		Можно удалить
GetDueCardRegAsync	string		Получение due картотеки регистрации
UserHasFileRightsAsync	bool	CardRight right	Право на файл
CanAddFileAsync	bool		Есть право добавлять файлы
CanEditInCurrentCardAsync	bool	CardRight right	Можно редактировать РК в текущей картотеке
CanEditAsync	bool		Есть право редактирования
CanMarkSendAsync	bool		
SendingRegistryFriendlyAsync	bool		Признак "Свой" для РК для централизованной отправки в ЦБ
SendingRegistryFriendlyForSndAsync	bool	IRefSend refSend	Признак "свой" для адресата

* **примечание:** последний параметр всех async методов - cancellationToken.

Н.5 Класс ResolutionSecurityService

Данный класс получаем через DataContextHelperService, метод DataContextHelper.ResolutionSecurity(Resolution, DocRc).

Список публичных методов (см. Таблица Н. 5):

Таблица Н. 5

Наименование	Возвращает	Параметры	Описание
--------------	------------	-----------	----------

Наименование	Возвращает	Параметры	Описание
CanAddResolutionAsync	bool		Проверка наличия прав на добавление поручения
CanAddResolutionOrProjectAsync	bool		Проверка наличия прав на добавление поручения или проекта поручения
CanReadAsync	bool		Проверка наличия прав на работу с поручением

* **Примечание:** последний параметр всех async методов - cancellationToken.

Приложение П

Сервисы кэширования

Список доступных сервисов кэширования (см. Таблица П. 1):
Таблица П. 1

Наименование	Тип	Описание
MemoryCacheService	Base	Сервис для кэширования любой таблицы, посредством Generic метода - GetOrCreateTableAsync
DocgroupCachingService	MultiObject	Сервис кэширования групп документов, включая значения по умолчанию
EosObjectCachingService	SingleObject	Сервис кэширования таблицы EOS_OBJECT
SysParamsCachingService	SingleObject	Сервис кэширования таблицы USER_PARMS с ISN_USER_OWNER = -99 и PARM_GROUP = -99
UserCachingService	MultiObject	Сервис кэширования пользователей и всех связанных таблиц, например USERDEP, USER_TECH и т.д.
DbModelCachingService	SingleObject	Сервис кэширования структуры базы данных, для получения различных полезных свойств, например длина колонки по имени таблицы и колонки.
ArDescriptCachingService	SingleObject	Кэширует таблицу ArDescript
DepartmentCachingService	MultiObject	Кэширует таблицу Department + возвращает связку с организацией через контакт.
CalendarCachingService	SingleObject	Кэширует таблицу calendarCls и предоставляет класс для вычисления плановых дат и т.д.

Приложение Р

Сервис DataContextHelperService. Методы

Список методов (см. Таблица Р. 1):

Таблица Р. 1

Наименование	Параметры	Описание	Пример использования
GetEntityAsync	CacheUse cacheUse, params (string propertyName, object value)[] conditions	Находит сущность по ряду условий с логическим "и" и с использованием локального кэша	<pre>var dgFileConstraintTop = await _dataContextHelper.GetEntityAsync<IDgFileConstraint>(cancellationToken, (nameof(IDgFileConstraint.IsnDocgroup), 0), (nameof(IDgFileConstraint.Category), 1));</pre>
GetEntityByKeyAsync	string key, object value, CacheUse cacheUse	Находит сущность по колонке и ключу с использованием локального кэша	<pre>var department = await _dataContextHelper.GetEntityByKeyAsync<IDepartment>(nameof(IDepartment.IsnNode), role.IsnPerson, cancellationToken);</pre>
GetEntityByPrimaryKeyAsync	object value, CacheUse cacheUse	Находит сущность по первичному ключу с использованием локального кэша	<pre>var prjRc = await _dataContextHelper.GetEntityByPrimaryAsync<IPrjRc>(prjVisaSign.IsnPrj, cancellationToken);</pre>
GetEntityFromReferenceAsync	string referenceName, IKeyedEntity sourceEntity, CacheUse cacheUse	Находит сущность по объекту и имени свойства с использованием локального кэша	<pre>var parentDep = await dataContextHelper.GetEntityFromReferenceAsync<IDepartment>(nameof(department.ParentNode), department, cancellationToken);</pre>
GetEntityFromReferenceAsync (extension-метод)	T1 parentObj, Expression<Func<T1, T2>> ex, CacheUse cacheUse = CacheUse.Normal	Находит сущность по объекту и имени свойства с использованием локального кэша	<pre>var parentDep = await dataContextHelper.GetEntityFromReferenceAsync(department, p => p.ParentNode, cancellationToken);</pre>
GetCollectionAsync	string key, string objName, string parentObjKey, IKeyedEntity parentObj, bool writeSingleObjectToCache, CacheUse cacheUse, string kindName, object kind, bool isObjNameCollection	Находит определенную коллекцию родительского объекта	<pre>var userCardList = await dbHelper.GetCollectionAsync<IUserCard>(nameof(IUserCard.IsnLclassif), nameof(IUserCard.UserCl), nameof(IUserCl.IsnLclassif), user, false, cancellationToken, cacheUse);</pre>
GetCollectionAsync	T1 parentObj,	Находит	<pre>var userCardList = await</pre>

Наименование	Параметры	Описание	Пример использования
(extension-метод)	Expression<Func<T1, IEnumerable<T2>>> ex, bool writeSingleObjectToCache = false, CacheUse cacheUse = CacheUse.Normal	определенную коллекцию родительского объекта	dbHelper.GetCollectionAsync(user, p => p.UserCards, cancellationTokens, false, cacheUse);
GetClassifFromUserList	long isnList, string isnName, bool skipDeleted, long? isnNode, CacheUse cacheUse	Находит коллекцию через таблицу UserLists	var departmentList = await dbHelper.GetClassifFromUserList<IDepartment>(isnList, nameof(IDepartment.IsnNode), true, isnNode, cancellationTokens: cancellationTokens);
GetTableAsync	CacheUse cacheUse	Находит таблицу с использованием локального кэша.	var buffFolders = await dch.GetTableAsync<IBuffFolder>(cancellationTokens);
GetOrCreateUserRightsHelperService		Возвращает класс для проверки прав пользователей.	var userRightsHelperService = dch.GetOrCreateUserRightsHelperService();
GetOrCreatePrjRcSecurityService	IPrjRc prjRc	Возвращает класс для проверки прав для РКПД	var prjRcSecurityService = dch.GetOrCreatePrjRcSecurityService(PrjRc);
GetOrCreateDocRcSecurityService	IDocRc docRc	Возвращает класс для проверки прав для РК	var docRcSecurityService = dch.GetOrCreateDocRcSecurityService(DocRc);
GetOrCreateResolutionSecurityService	IResolution resolution	Возвращает класс для проверки прав для резолюции	var resolutionSecurityService = dch.GetOrCreateResolutionSecurityService(resolution);
GetOrCreatePrjRcHelperService	IPrjRc prjRc	Возвращает класс для вспомогательных методов для РКПД	var prjRcHelperService = dch.GetOrCreatePrjRcHelperService(PrjRc);
GetOrCreateDocRcHelperService	IDocRc docRc	Возвращает класс для вспомогательных методов для РК	var docRcHelperService = dch.GetOrCreateDocRcHelperService(DocRc);
GetOrCreateBufFolderSecurityService	IBufFolder bufFolder	Возвращает класс для проверки прав для BufFolder	var bufFolderSecurityService = dch.GetOrCreateBufFolderSecurityService(BufFolder);
GetOrCreateArValueHelperServiceAsync	IKeyedEntity entity, Docgroup docgroup	Возвращает класс для вспомогательных методов для доп. реквизитов	var arValueHelperService = await dch.GetOrCreateArValueHelperServiceAsync(entity, docgroup, cancellationTokens);

Наименование	Параметры	Описание	Пример использования
GetSysParmsAsync		Возвращает класс SysParms системных настроек.	<pre>var sysParms = await dch.GetSysParmsAsync(cancellationToken);</pre>

*** примечание:** во всех async методах присутствует параметр CancellationToken

Список методов расширения для DataContextHelperService (находится в Eos.Delo.Platform.Storage.Extensions) (см. Таблица П. 2):

Таблица П. 2

Описание
CheckFeedWorksAsync(System.Threading.CancellationToken)
Contact2DepartmentAsync(Eos.Delo.Platform.Storage.Model.IContact, System.Threading.CancellationToken)
DeleteClassifAsync(string, string, string, System.Threading.CancellationToken)
DepMoveToCabinetAsync(string, long, System.Threading.CancellationToken)
EntityHasDoubleValueAsync(Microsoft.EntityFrameworkCore.ChangeTracking.EntityEntry, System.Linq.IQueryable<Eos.Platform.Storage.IKeyedEntity>, string, object, System.Threading.CancellationToken, string[])
EntityHasDoubleValueAsync(Microsoft.EntityFrameworkCore.ChangeTracking.EntityEntry, System.Linq.IQueryable<Eos.Platform.Storage.IKeyedEntity>, System.Collections.Generic.IEnumerable<(string param, bool equal, object value)>, System.Threading.CancellationToken)
EntityHasDoubleValueAsync(Microsoft.EntityFrameworkCore.ChangeTracking.EntityEntry, System.Linq.IQueryable<Eos.Platform.Storage.IKeyedEntity>, System.Threading.CancellationToken, string[])
EntityHasDoubleValueAsync<T>(Microsoft.EntityFrameworkCore.ChangeTracking.EntityEntry, string, object, System.Threading.CancellationToken, string[])
EntityHasDoubleValueAsync<T>(Microsoft.EntityFrameworkCore.ChangeTracking.EntityEntry, System.Threading.CancellationToken, string[])
EosJpRunAsync(string, System.Threading.CancellationToken)
GetContactListItemsAsync(long, System.Threading.CancellationToken)
GetDateFromDocgroupAsync(string, System.DateTime, System.Threading.CancellationToken)
GetDateFromOrganizAsync(string, System.DateTime, System.Threading.CancellationToken)
GetDepartmentListItemsAsync(long, long?, System.Threading.CancellationToken)
GetDocgroupCIWithDocnumberFlagAsync(Eos.Delo.Platform.Storage.Model.IDocgroupCI, System.Threading.CancellationToken)

Описание
GetDocgroupListItemsAsync(long, System.Threading.CancellationToken)
GetNum2Async(long, int, System.Threading.CancellationToken)
GetParentPrjVisaSignAsync(Eos.Delo.Platform.Storage.Model.IPrjVisaSign, System.Threading.CancellationToken)
GetPareRefLinkAsync(Eos.Delo.Platform.Storage.Model.IRefLink, System.Threading.CancellationToken)
GetPrintFileFormatListAsync(Eos.Delo.Platform.Storage.Model.IPrintForm, System.Threading.CancellationToken, Eos.Delo.Platform.Storage.Services.CacheUse)
GetPrjNum2Async(long, int, System.Threading.CancellationToken)
GetRefFileListAsync(Eos.Platform.Storage.IKeyedEntity, string, long, System.Threading.CancellationToken, Eos.Delo.Platform.Storage.Services.CacheUse)
GetMainRefFileListAsync(Eos.Platform.Storage.IKeyedEntity, string, long, System.Threading.CancellationToken, Eos.Delo.Platform.Storage.Services.CacheUse)
GetRefLinkListAsync(Eos.Platform.Storage.IKeyedEntity, string, long, System.Threading.CancellationToken, Eos.Delo.Platform.Storage.Services.CacheUse)
GetRubricCilListItemsAsync(long, System.Threading.CancellationToken)
GetSeqAsync(string, System.Threading.CancellationToken)
GetUserCardListAsync(Eos.Delo.Platform.Storage.Model.IUserCl, System.Threading.CancellationToken, Eos.Delo.Platform.Storage.Services.CacheUse)
GetUserDepListAsync(Eos.Delo.Platform.Storage.Model.IUserCl, System.Threading.CancellationToken, Eos.Delo.Platform.Storage.Services.CacheUse)
GetUserRequestListAsync(Eos.Delo.Platform.Storage.Model.ISrchRequest, System.Threading.CancellationToken, Eos.Delo.Platform.Storage.Services.CacheUse)
JoinClassifAsync(string, long, long, System.Threading.CancellationToken)
KillDbSessionAsync(int, System.Threading.CancellationToken)
MarkReadProtAsync(string, long, System.Threading.CancellationToken)
PimpImpersonateAsync(long, System.Threading.CancellationToken)
PrjSaveAsync(long, System.Threading.CancellationToken)
RcSaveAsync(Eos.Delo.Platform.Storage.Model.IDocRc, string, long, string, System.Threading.CancellationToken)

Описание
ReserveDocNumAsync(string, int, string, string, System.Threading.CancellationToken)
ReservePrjNumAsync(string, int, string, System.Threading.CancellationToken)
ReturnNumAsync(string, long, long, string, string, System.Threading.CancellationToken)
ReturnPrjNumAsync(string, long, long, string, string, System.Threading.CancellationToken)
UpdJournalOwnerAsync(long, System.Threading.CancellationToken)
WriteProtAsync(string, string, string, string, long, string, string, long?, System.Threading.CancellationToken)

Приложение С

Перечень доступных UOD и их параметры (UOD_PROP) для составления отчетных форм

Перечень доступных UOD и их параметры (UOD_PROP) для составления отчетных форм (см. Таблица С. 1):

Таблица С. 1

Тип	UOD	UOD_PROP	Дополнительно
Дата	un_period_date_srch	set_options() cut_options() required(true)	Если в списке операций остается один вариант - список не отображается
	uod_year	required(true)	-
	uod_time	required(true)	-
Текст	uod_text_option	required(true)	-
	uod_num	rb_data(val()) required(true)	Если rb_data(val()) не задан, является простым текстовым полем
	uod_context	required(true)	-
Число	uod_em_rb	options(boolean) min(number) max(number) step(number) float(number) default_opt(number) required(true)	Если options не задан, считается что options = true step - шаг по клику на +/- float - с точкой (значение = количество символов после точки) default_opt - устанавливает опцию (из списка) по умолчанию
Справочник	uod_classif	title(string) source(string) return_due(boolean) select_multi(false) select_leafs(boolean) select_nodes(boolean) skip_deleted(boolean) representative(boolean) required(true)	-
	uod_multi	title(string) list(string) required(true)	-
	uod_journal_recincard	title(string) source(string) rb_data(val()) return_due(boolean) select_multi(false) select_leafs(boolean)	rb_data(val()) обязателен

Тип	UOD	UOD_PROP	Дополнительно
		select_nodes(boolean) skip_deleted(boolean) representative(boolean) required(true)	
Выбор значения	uod_dddw	rb_data(val()) multi(boolean) checkable(true) required(true)	Если multi не задан, считается что multi = false checkable - добавляет в список чекбоксы
	un_cbx	rb_data(val()) required(true)	-
	uod_checkboxgroup	rb_data(val()) required(true)	-

Если форма проходит валидацию - критерии с параметром required(true) должны быть обязательно заполнены

Параметр rb_data(val()) задает список опций для некоторых контролов:

rb_data(string) - принимает строку (разделитель ";"), которая является перечнем отображаемых в контроле опций (например - rb_data(;Вариант1;Вариант2))

val(string) - принимает строку (разделитель ";"), которая является перечнем значений, которые соответствуют опциям из rb_data() (например - val(;1;2))

итого при передаче параметра вида rb_data(;Вариант1;Вариант2)val(;1;2), например, в uod_num получим список опций , где:

пусто = null

Вариант1 = 1

Вариант2 = 2

Параметры set_options и cut_options предназначены для изменения стандартного списка операций:

set_options - устанавливает новый список операций, содержащий перечисленные в нем операции

cut_options - исключает из стандартного списка перечисленные в нем операции
операции перечисляются в произвольном порядке и разделяются ";", например set_options(equal;range)

Перечень доступных операций контроля даты:

- range - В диапазоне дат
- equal - Совпадает с
- today - Сегодня
- todayN - Сегодня + N
- beforeN - Сегодня + N и ранее
- afterN - Сегодня + N и позднее
- curWeek - Текущая неделя
- curMonth - Текущий месяц
- curQuarter - Текущий квартал
- curYear - Текущий год

- prevWeek - Прошедшая неделя
- prevMonth - Прошедший месяц
- prevQuarter - Прошедший квартал
- prevYear - Прошедший год
- week - Последняя неделя
- month - Последний месяц
- quarter - Последний квартал
- year - Последний год
- notInRange - Вне диапазона дат
- isNotNull - Есть значение
- isNull - Нет значения

Перечень значений для операций числового контроля:

- 1 - равно
- 2 - больше
- 3 - меньше
- 4 - больше или равно
- 5 - меньше или равно
- 6 - в диапазоне
- 7 - не равно
- 8 - вне диапазона

Параметры для справочников:

- title - заголовок критерия
- source - название справочника (как правило, значение параметра совпадает с именем таблицы в БД, в которой хранятся записи справочника. Например: source (DEPARTMENT), source (RUBRIC_CL) и т.п.)
- return_due - тип идентификатора записи справочника (если у справочника два идентификатора этот параметр определяет который из них возвращать)
- select_multi - одиночный или множественный выбор записей
- select_leafs - для выбора из справочника разрешены листья (по умолчанию true)
- select_nodes - для выбора из справочника разрешены вершины (по умолчанию true)
- skip_deleted - отображение логически удаленных элементов
- representative - отображение только представителей организации (только для справочника CONTACT)