



ТОМ 3

Руководство разработчика

Редакция 12 от 23.01.2018



Содержание

Глоссарий	3
Введение	5
1. API Системы MODES-Centre	6
1.1. <i>Функции API</i>	7
1.2. <i>События</i>	12
1.3. <i>Описание объектов</i>	12
1.4. <i>Пример получения данных</i>	21
2. Тестовое приложение библиотеки ModesApiExternal.dll	22
2.1. <i>Получение среза данных</i>	23
2.2. <i>Работа со слоями данных</i>	24
2.3. <i>Получение значений переменных характеристик</i>	26
2.4. <i>Работа с макетами</i>	28
2.5. <i>Плановые графики</i>	29
3. Редактор локальных правил	31
3.1. <i>Интерфейс Редактора локальных правил</i>	31
3.2. <i>Редактирование локальных библиотек</i>	32
3.3. <i>Публикация локальных библиотек</i>	39
4. Хранимые процедуры	40
4.1. <i>Процедура GetPlanGraph</i>	40
4.2. <i>Процедура GetPlanVersion</i>	40
4.3. <i>Процедура GetVarValuesByEqType</i>	41
4.4. <i>Процедура GetVarValuesByEquipments</i>	42
4.5. <i>Процедура GetVarValuesByStation</i>	43
4.6. <i>Процедура GetVarValuesSlice</i>	44
Перечень иллюстраций	46

Глоссарий

АСПУ	Автоматизированная система передачи уведомлений
БД	База данных
ВСВГО	Выбор состава включённого генерирующего оборудования
ГТП	Группа точек поставки
ГТПГ	Группа точек поставки генерации
ГТПП	Группа точек поставки потребления
ДГ	Диспетчерский график
ЕГО	Единица генерирующего оборудования
ЕКО	Единица котельного оборудования
ИС СО	Информационная система Системного Оператора
КИСУ	Консоль сбора данных об изменении системных условий
КИТС	Корпоративная интеграционно-транспортная система АО «СО ЕЭС»
МППМ	Подсистема «Модуль проверки правил и маршрутизации сообщений»
ОДУ	Объединённое диспетчерское управление
ОРЭМ	Оптовый рынок электрической энергии и мощности
ПГАД	Подсистема гарантированной асинхронной доставки
ПДГ	Прогнозный диспетчерский график
РГЕ	Режимная генерирующая единица
РДУ	Региональное диспетчерское управление
РСВ	Рынок «на сутки вперёд»
САШ	Программное обеспечение «АРМ администратора ШЛЮЗ»

СШВС	WEB интерфейс для администрирования ШЛЮЗ (IBM Data power)
Шлюз СО	Программно-аппаратный шлюз АО «СО ЕЭС» для информационного взаимодействия с внешними организациями и системами
ЭЦП	Электронная цифровая подпись
SOA	Service-Oriented Architecture – Сервис-ориентированная архитектура
XML	Extensible Markup Language – расширяемый язык разметки
XSD	XML-schema-definition – язык описания структуры XML
WS СО	Web-сервисы, предоставляемые Системным Оператором

Введение

Техническая документация для MODES-Centre включает в себя 3 тома:

Том 1 – Руководство администратора;

Том 2 – Руководство пользователя;

Том 3 – Руководство разработчика.

Данный том документации содержит описание настройки и работы **Тестового приложения библиотеки ModesApiExternal.dll**, представляющего собой наглядный пример использования библиотеки **ModesApiExternal.dll** сторонними программно-аппаратными комплексами для подключения к системе **MODES-Centre** (далее Система).

1. API Системы MODES-Centre

Система создана на основе технологии **WCF** (Windows Communication Foundation), обеспечивающей удалённое взаимодействие клиентов Системы и сервиса централизованной обработки данных.

Для программного высокоуровневого доступа к Системе предназначен набор функций **API**, предоставляемых библиотекой **ModesApiExternal.dll**, описание которой приведено ниже.

Библиотека **ModesApiExternal.dll** предназначена для разработки приложений с использованием *Microsoft .NET Framework*. Разработка нового клиентского приложения должна вестись в следующем виде:

1. Приложение необходимо разрабатывать, используя *.Net FrameWork* версии 4.0 или выше.
2. В проект необходимо добавить две библиотеки **ModesApiExternal.dll** и **Infrastructure.dll**.

Порядок работы с Системой:

- Для начала работы необходимо инициализировать подключение к сервису **MODES**:

```
ModesApiFactory.Initialize(serviceHost);
```

где *serviceHost* – это строка подключения к сервису **MODES**, например: *modes-cdu*, представляющая собой сетевое имя компьютера.

- После успешной инициализации необходимо получить интерфейс доступа к Системе:

```
IApiExternal api = ModesApiFactory.GetModesApi();
```

- После получения объекта **API MODES** вся дальнейшая работа ведётся через него и получаемые через него объекты.
- После окончания работы с сервисом **MODES-Centre** для завершения сессии и освобождения ресурсов необходимо вызвать функцию закрытия соединения:

```
ModesApiFactory.CloseConnection();
```

Основополагающим объектом при работе с API MODES является объект типа `IGenObject`. Этот объект представляет единицу энергооборудования. Он всегда содержит условно-постоянную информацию, характеризующую это оборудование, и может содержать данные по условно-переменным параметрам (могут быть загружены и выгружены). Данные по условно-переменным параметрам для выбранного объекта всегда загружаются по всем параметрам и всем слоям, но в каждый момент времени получение данных (функция `GetValue`) производится только из выбранного слоя. Выбор слоя, из которого происходит получение данных, производится функцией `ResetValuesToLayer` (см. [глава 1.3](#)).

1.1. Функции API

Функция получения дерева оборудования на выбранную дату:

```
IModesTimeSlice GetModesTimeSlice(DateTime dt, SyncZone syncZone, TreeContent treeContent, Boolean loadData);
```

- `dt` – дата начала суток по московскому времени в формате UTC;
- `syncZone` – синхронная зона;
- `treeContent` – содержимое дерева оборудования (все объекты, объекты подготовки данных, объекты плановых графиков);
- `loadData` – управление загрузкой данных (`true` – загружать данные, `false` – прочесть дерево энергообъектов без загрузки данных по ним).

Функция возвращает параметр типа `IModesTimeSlice`, который содержит дерево оборудования и значения параметров на сутки.

Для получения значения параметров энергооборудования на различные сутки достаточно один раз получить дерево оборудования функцией `GetModesTimeSlice` и затем подгружать/обновлять данные по выбранным объектам в пределах актуальности метаданных (свойства `DtFrom`, `DtTo` объекта `IModesTimeSlice`) вызывая метод `RefreshGenObjects`. При переходе к периоду действия новых метаданных необходимо перечитать дерево оборудования.

Функция получения/обновления данных по выбранным объектам:

```
void RefreshGenObjects(IList<IGenObject> genObjs, DateTime  
dt, SyncZone syncZone);
```

- `genObjs` – список объектов, для которых необходимо выполнить получение/обновление данных;
- `dt` – дата, на которую необходимо выполнить получение/обновление данных. Значение параметра должно представлять начало суток по московскому времени в формате UTC;
- `syncZone` – синхронная зона.

При выполнении этой функции в объекты, переданные в неё, загружается информация по всем слоям данных.

Функция получения/обновления данных по выбранным объектам указанного слоя:

```
void RefreshGenObjects(IList<Modes.BusinessLogic.IGenObject>  
genObjs, DateTime dt, int source, int layerOffset,  
ModesTaskType task, SyncZone syncZone);
```

- `genObjs` – список объектов, для которых необходимо выполнить получение/обновление данных;
- `dt` – дата, на которую необходимо выполнить получение/обновление данных. Значение параметра должно представлять начало суток по московскому времени в формате UTC;
- `source` – при значении 0 - данные станции, при значении 1 – данные СО;
- `syncZone` – синхронная зона.
- `task` – этап планирования (свойство `IdTask` объекта `ModesLayer`), к которому относится слой.
- `layerOffset` – смещение выбранного слоя данных (свойство `Offset` объекта `ModesLayer`);

При выполнении этой функции в объекты, переданные в неё, загружается информация по выбранному слою данных.

Функция получения актуального слоя данных:

```
ModesLayer GetCurrentLayer(DateTime verifyDate, DateTime  
dayPlanning, SyncZone syncZone);
```

- `verifyDate` – дата и время, на которое необходимо узнать актуальный слой, например: текущее время. Параметр должен представлять московское время в формате UTC;
- `dayPlanning` – сутки планирования, на которые необходимо узнать актуальный слой. Значение параметра должно представлять начало суток по московскому времени в формате UTC;
- `syncZone` – синхронная зона.

Функция возвращает объект типа `ModesLayer` содержащий информацию об актуальном слое на сутки планирования `dayPlanning` в момент времени `verifyDate`.

Функция получения списка всех слоёв данных для выбранной синхронной зоны:

```
IList<ModesLayer> GetLayersBySyncZone(SyncZone syncZone)
```

- `syncZone` – синхронная зона.

Функция возвращает список всех слоёв, зарегистрированных в Системе для синхронной зоны `syncZone`.

Функция получения списка описаний макетов данных 53500:

```
IList<IMaketHeader> GetMaketHeaders53500(DateTime dt1,  
DateTime dt2);
```

- `DateTime dt1` – начало диапазона;
- `DateTime dt2` – конец диапазона.

Функция возвращает список описаний макетов данных 53500 в указанном интервале.

Функция получения списка описаний макетов данных 53500 по уникальным идентификаторам:

```
IList<IMaketHeader> GetMaketHeaders53500(Guid[] maketsUid);
```

- Guid[] maketsUid – массив уникальных идентификаторов макетов.

Функция возвращает список описаний макетов данных 53500 по уникальным идентификаторам.

Функция получения списка данных оборудования макетов 53500 по уникальным идентификаторам:

```
IList<IMaketEquipment> GetMaket53500Equipment(IList<Guid> mrid)
```

- IList<Guid> mrid – список уникальных идентификаторов макетов.

Функция возвращает список данных оборудования макетов 53500 по уникальным идентификаторам.

Функция получения перечня пакетов СДК по станции:

```
GetSdcPackRaw(var obj, DateTime dt1, DateTime dt2);
```

- var obj – идентификатор станции;
- DateTime dt1 – начало диапазона;
- DateTime dt2 – конец диапазона.

Функция возвращает перечень пакетов СДК по указанной станции в рамках заданного интервала.

Функция получения массива плановых характеристик:

```
IList<PlanFactorItem> GetPlanFactors()
```

Функция получения версий плановых графиков, находящихся в хранилище на указанную дату, по указанному оборудованию (по станции, которой оно принадлежит):

```
IList<PlanInfoItem> GetPlanVersionsByDay(DateTime dt,
IGenObject Eq)
```

- DateTime dt - дата начала суток по Московскому времени в формате UTC;
- IGenObject Eq - объект, для которого возвращаются имеющиеся в системе версии плановых графиков.

Функция получения значений плановых графиков в интервале от 1 часа по 24 включительно указанных суток планирования:

```
IList<PlanValueItem> GetPlanByDay(DateTime dt,
IList<PlanType> pTypeList, IList<IGenObject> lstGenObjects)
```

- DateTime dt - целевые сутки планирования;
- IList<PlanType> pTypeList - список типов запрашиваемых планов;
- IList<IGenObject> lstGenObjects - оборудование, по которому запрашиваются планы.

Функция получения актуального планового графика в указанном диапазоне:

```
IList<PlanValueItem> GetPlanValuesActual(DateTime dtFrom,
DateTime dtTo, IGenObject genObject)
```

- DateTime dtFrom - начало интервала;
- DateTime dtTo - конец интервала;
- IGenObject genObject - объект, для которого возвращаются актуальный план для каждых суток в заданном интервале.

1.2. События

Объект типа `IApiExternal` содержит событие, на которое могут быть подписаны клиентские приложения:

- `OnData53500Modified` – событие возникает в момент изменения данных по энергооборудованию модели M53500. Параметры события содержат дату, на которую произошли изменения, а также идентификаторы объектов;
- `OnMaket53500Changed` – событие возникает в момент изменения данных по макетам 53500. Параметры события содержат количество изменённых макетов и уникальные идентификаторы изменённых макетов.
- `OnPlanDataChanged` – событие возникает в момент получения планового графика. Параметры события содержат наименование станции, сутки планирования, на которые получен план, и новый план.

1.3. Описание объектов

Описание объекта `IModesTimeSlice`:

- `DateTime` `Date` – сутки, на которые получен срез данных;
- `DateTime?` `DtFrom`, `DateTime?` `DtTo` – диапазон, в котором действует данный срез без изменения метаданных;
- `SyncZone` `SyncZone` – синхронная зона, которой принадлежит данный срез данных;
- `ModelType` `SliceModelType` – тип модели (53500);
- `ICheck` `Checker` – интерфейс проверки данных;
- `ICalc` `Calculator` – интерфейс дорасчёта данных;
- `IAutoFill` `AutoFill` – интерфейс автозаполнения;
- `ICheck` `LocalChecker` – локальный интерфейс проверки данных;
- `ICalc` `LocalCalculator` – локальный интерфейс дорасчёта данных;
- `IAutoFill` `LocalAutoFill` – локальный интерфейс автозаполнения;
- `ReadOnlyCollection<IGenObject>` `GenTree` – дерево оборудования.

Интерфейс IMaketHeader:

- `Guid Mrid` – уникальный идентификатор;
- `DateTime? DtCreated` – дата создания;
- `DateTime? DtReceived` – дата получения;
- `DateTime? DtSent` – дата отправки;
- `DateTime DtTarget` – дата, на которую ориентированы данные макета;
- `int State` – состояние;
- `int Id_Task` – идентификатор типа планирования;
- `bool? AfterGateClosed` – признак получения макета после закрытия ворот;
- `string Sender` – отправитель;
- `string Comment` – комментарий;
- `bool? CheckPassed` – признак прохождения проверки (`true` – проверка пройдена, `false` – проверка не пройдена);
- `bool? SigValid` – признак актуальности подписи (`true` – подпись актуальна, `false` – не актуальна, `null` – подпись отсутствует);
- `int? Id_Layer` – идентификатор слоя планирования;
- `int? Id_AcceptLayer` – идентификатор слоя планирования, на который акцептован макет;
- `int IdArea` – идентификатор зоны ответственности;
- `bool IsStationMaket` – признак станционной принадлежности макета;
- `StatePosted StPosted` – почтовое состояние макета (отправлен, не отправлен и т.д.);
- `StateProceed StProceed` – состояние макета (акцептован, не акцептован и т.д.).

Интерфейс `IMaketEquipment`:

- `ReadOnlyCollection<IGenObject> GenTree` – дерево оборудования – данные макета;
- `Guid Mrid` – идентификатор макета.

Дерево оборудования, находящееся в срезе данных, включает в себя объекты типа `IGenObject`, которые представляют собой энергообъекты, содержащие условно-постоянные и условно-переменные характеристики.

Описание объекта `IGenObject`:

- `int IdInner` – уникальный идентификатор объекта (среди объектов данной модели `ModelType`) в базе данных MODES (только чтение);
- `string Name` – наименование объекта (короткое имя объекта) – загружается из внешних систем, в которых может выступать в качестве идентификатора, например: идентификатор станции (только чтение);
- `string Description` – текстовое описание/наименование объекта (только чтение);
- `ReadOnlyCollection<IGenObject> Children` – список потомков (только чтение);
- `IGenObject Owner` – объект-родитель (только чтение);
- `IConstParamCollection ConstParams` – условно-постоянные характеристики (только чтение);
- `ReadOnlyCollection<IVarParam> VarParams` – условно-переменные характеристики (только чтение);
- `void UnloadData()` – выгрузка данных по параметрам объекта (освобождает память, используемую для хранения значений условно-переменных характеристик);
- `void SetVarParametersToTask(ModesTaskType task, ValueSource? source = null)` – установка параметров, соответствующих этапу планирования;

- `bool` `IsModified` – признак наличия изменений в параметрах объекта (только чтение);
- `IGenObjectType` `GenObjType` – тип объекта (только чтение);
- `ReadOnlyCollection<CheckResult>` `CheckResults` – список результатов проверок для данного объекта (только чтение);
- `void` `SetCheckResult(IVarParam property, int i, string message, CheckResultType checkResultType, int checkCode, bool onlySoCheck = false)` – установить результат проверки (используется в библиотеке проверок);
- `bool` `HasError` – признак наличия в объекте ошибок в данных (только чтение);
- `bool` `HasWarning` – признак наличия в объекте предупреждений (только чтение);
- `int` `MaxPointCount` – количество интервалов (только чтение);
- `bool` `NeedsCalc` – признак необходимости дорасчёта;
- `bool` `NeedsCheck` – признак необходимости проверок;
- `void` `ClearCheckResults()` – очистка результатов проверки для объекта;
- `IConstParam` `GetConstParam(string paramName)` – получение условно-постоянного параметра по имени;
- `IConstParam` `GetConstParam(int id)` – получение условно-постоянного параметра по идентификатору;
- `IVarParam` `GetVarParam(string paramName)` – получение условно-переменного параметра;
- `IModesTimeSlice` `TimeSlice` – временной срез к которому принадлежит объект (только чтение);
- `int` `Id` – идентификатор объекта в БД (только чтение);

- `void ResetValues()` – сбросить отредактированные значения к исходному состоянию объекта;
- `void ResetValuesToLayer(int layerOffset, ValueSource source)` – сбросить отредактированные значения к исходному состоянию объекта в указанном слое;
- `int GridStep` – шаг изменения сетки параметров;
- `int Order` – порядок отображения объектов;
- `bool DataLoaded` – признак того, что данные загружены;
- `bool HasData(int layerOffset, ValueSource source)` – проверка наличия данных по указанному слою планирования, от заданного источника данных.

Коллекции условно-переменных и условно-постоянных характеристик содержат экземпляры характеристик `IVarParam` и `IConstParam` соответственно, которые используются для получения значений параметров.

Описание объекта `IVarParam`:

- `int Step` – шаг изменения параметра сек. (только чтение);
- `bool IsInput` – признак параметра, вводимого вручную (только чтение);
- `bool IsRequired` – признак обязательного параметра (только чтение);
- `object GetValue(int i)` – получение значения параметра;
- `object GetValue(DateTime time)` – получение значения параметра;
- `int PointCount` – количество временных интервалов для параметра (только чтение);
- `bool IsModified(int i)` – признак наличия изменений в параметре в выбранном интервале;
- `object GetDefaultValue()` – получение значения по умолчанию;

- `bool HasChanges` – признак наличия изменений в параметре относительно значений в базе данных (только чтение);
- `PointInfo GetPointInfo(int i)` – получение дополнительной информации о значении в выбранном интервале;
- `PointInfo GetPointInfo(int i, ValueSource? sourceFilter, int? layerFilter)` – получение дополнительной информации о значении в выбранном интервале;
- `bool IsManuallyEdited(int i)` – признак ручного редактирования параметра в указанном интервале;
- `string GetComment(int i)` – получение комментария для указанного интервала;
- `void ClearEditValues()` – очистка слоя редактирования;
- `bool OnlySo` – признак принадлежности параметра только Системному Оператору (только чтение);
- `void ResetEditValuesToLayer(int offset, ValueSource valueSource)` – сброс значения изменённых характеристик до значений указанного слоя планирования, заданных указанным источником данных;
- `void ResetEditValues()` – сброс значения изменённых характеристик;
- `bool IsOuParam` – признак принадлежности параметра оперативным уведомлениям (только чтение);
- `bool IsRsvParam` – признак принадлежности параметра уведомлениям РСВ (только чтение);
- `bool IsVsvgoParam` – признак принадлежности параметра уведомлениям ВСВГО (только чтение);
- `bool IsLocal` – признак локального параметра (только чтение);
- `bool HasData(int layerOffset, ValueSource source)` – проверка наличия станционных данных в указанном слое.

Для выбора слоя, из которого необходимо получать данные, интерфейс `IGenObject` содержит метод:

```
void ResetValuesToLayer(int layerOffset, ValueSource source)
```

- `layerOffset` – смещение выбранного слоя данных (свойство `Offset` объекта `ModesLayer`);
- `source` – признак источника данных (позволяет выбрать слой СО или станции).

После вызова этого метода, вызов функции `GetValue` у экземпляров условно-переменных характеристик (`IVarParam`) коллекции `VarParams` будет возвращать значения параметров, соответствующие тому смещению (определяет слой данных) и источнику (определяет принадлежность к СО или станции), которые были переданы в метод `ResetValuesToLayer`.

Описание объекта `IMaketHeader`:

- `Guid Mrid` – уникальный идентификатор;
- `DateTime? DtCreated` – дата создания;
- `DateTime? DtReceived` – дата получения;
- `DateTime? DtSent` – дата отправки;
- `DateTime DtTarget` – дата, на которую ориентированы данные макета;
- `int State` – состояние;
- `int Id_Task` – идентификатор типа планирования;
- `bool? AfterGateClosed` – признак получения макета после закрытия ворот;
- `string Sender` – отправитель;
- `string Comment` – комментарий;

- `bool? CheckPassed` – признак прохождения проверки (`true` – проверка пройдена, `false` – проверка не пройдена);
- `bool? SigValid` – признак актуальности подписи (`true` – подпись актуальна, `false` – не актуальна, `null` – подпись отсутствует)
- `int? Id_Layer` – идентификатор слоя планирования
- `int? Id_AcceptLayer` – идентификатор слоя планирования на который акцептован макет
- `int IdArea` – идентификатор зоны ответственности
- `bool IsStationMaket` – признак того, что макет станционный
- `StatePosted StPosted` – почтовое состояние макета (отправлен, не отправлен и т.д.)
- `StateProceed StProceed` – состояние макета (акцептован, не акцептован и т.д.)

Интерфейс `IMaketEquipment`:

- `ReadOnlyCollection<IGenObject> GenTree` – дерево оборудования – данные макета;
- `Guid Mrid` – идентификатор макета.

`PlanFactorItem` – плановая характеристика:

- `int Id` – идентификатор характеристики;
- `String Name` – название характеристики;
- `String Description` – описание характеристики;
- `String DefaultColourName` – цвет по умолчанию.

`PlanInfoItem` – версия ПГ:

- `DateTime DT` – сутки планирования;
- `PlanType Type` – тип плана;

- `DateTime DTReceived` – дата получения версии плана.

`PlanValueItem` – значение характеристики в определённый момент времени:

- `string ObjName` – идентификатор объекта (имя);
- `int ObjType` – тип объекта;
- `int ObjFactor` – характеристики;
- `DateTime DT` – дата с точностью до часа;
- `Double Value` – значение;
- `PlanType Type` – тип плана.

1.4. Пример получения данных

```
// genObj объект типа IGenObject

foreach (var varParam in genObj.VarParams)
{

    Console.WriteLine(String.Format("Характеристика {0}", varParam.Name));

    for (int i = 0; i < varParam.PointCount; i++)
    {
        object value = varParam.GetValue(i);

        Console.WriteLine(String.Format("Интервал: {0}, значение: {1} ", i, value));
    }
}

foreach (var constParam in genObj.ConstParams)
{
    object value = constParam.GetValue();
    Console.WriteLine(String.Format("Характеристика {0}, значение {1}", constParam.Name, value));
}
```

2. Тестовое приложение библиотеки ModesApiExternal.dll

Тестовое приложение библиотеки ModesApiExternal.dll – ApiExternalTestApp.exe (далее **Тестовое приложение**), предназначено для иллюстрации работы и отображения её результатов при взаимодействии с библиотекой **ModesApiExternal.dll**.

Для работы клиентской части ПО MODES-Centre (в том числе и приложений, реализованных с помощью **ModesApiExternal.dll**) необходимо обеспечить доступность сервера MODES-Centre для клиентского ПО на всех АРМ пользователей по ЛВС, а именно:

1. Предоставить возможность обращения с клиентских АРМ по имени (или IP-адресу) сервера MODES-Centre.
2. Открыть порты между всеми существующими АРМ, на которых установлен клиент MODES-Centre, и сервером MODES-Centre:

Порт	Протокол	Описание
8081	net.tcp	Взаимодействие клиента с сервисом MODES-Centre
8082	http	Авторизация на сервисе MODES-Centre клиента средствами локальной аутентификации MODES-Centre
8085	http	Получение данных для автономной работы клиента
8086	net.tcp	Сервис генерации отчётов для клиента
8090	http	Выдача клиенту сертификата для подписания данных в канале к сервису MODES-Centre при авторизации на сервисе MODES-Centre клиента средствами аутентификации MODES-Centre

После запуска **Тестового приложения** элементы главного окна заблокированы. Для начала работы необходимо подключиться к сервису MODES, для чего следует воспользоваться пунктом системного меню **Подключиться**.

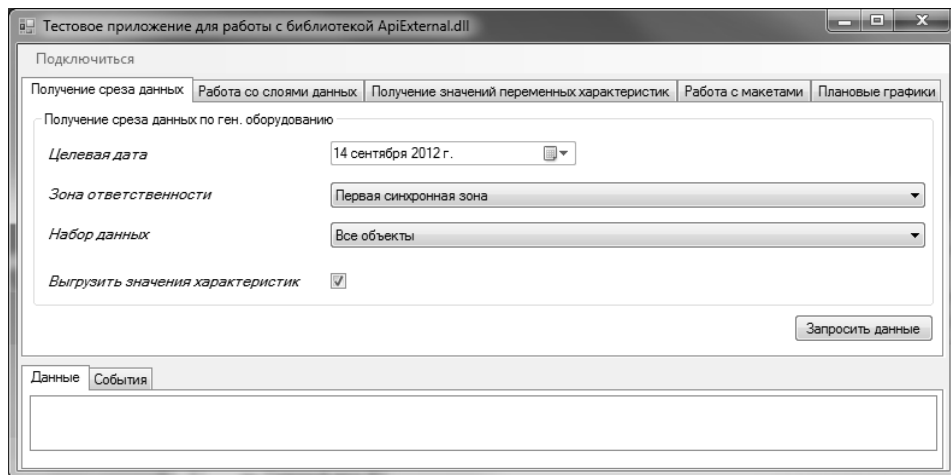


Рис. 1. Главное окно тестового приложения

В появившемся окне подключения необходимо ввести имя сервера, к которому необходимо подключиться и нажать кнопку **ОК**. Если подключение произведено успешно, то элементы главного окна будут разблокированы.

Главное окно **Тестового приложения** состоит из 2-х частей:

- панель с вкладками для задания параметров запроса к БД, описание которых приведено в следующих главах;
- панель для вывода результатов запроса и отображения событий системы.

2.1. Получение среза данных

Первая вкладка панели задания параметров запроса – **Получение среза данных**. Результат запрос представляет собой срез данных по всему оборудованию указанного типа в указанной синхронной зоне на указанную дату. Активированная опция **Выгрузить значения характеристик** позволяет выгружать данные по переменным характеристикам оборудования. Из выпадающего списка **Набор данных** можно выбрать, какие объекты следует выгружать:

- все объекты;
- объекты для интерфейсов подготовки данных;
- объекты для интерфейсов плановых графиков.

После нажатия кнопки **Запросить данные** в панели вывода результатов запроса отобразятся объекты с указанием их имён, наименований и типов.

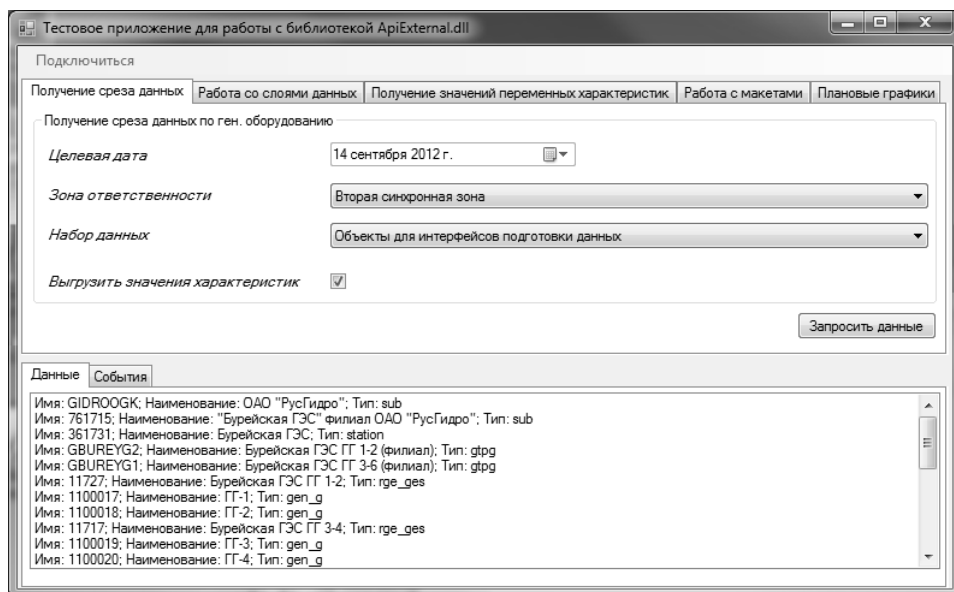


Рис. 2. Отображение результата запроса среза данных

2.2. Работа со слоями данных

Вкладка **Работа со слоями данных** позволяет получить актуальный слой указанных суток планирования относительно целевой даты по указанной синхронной зоне. Также можно получить весь список слоёв данных по указанной зоне ответственности.

Результат запроса актуального слоя на 21.09.2011:

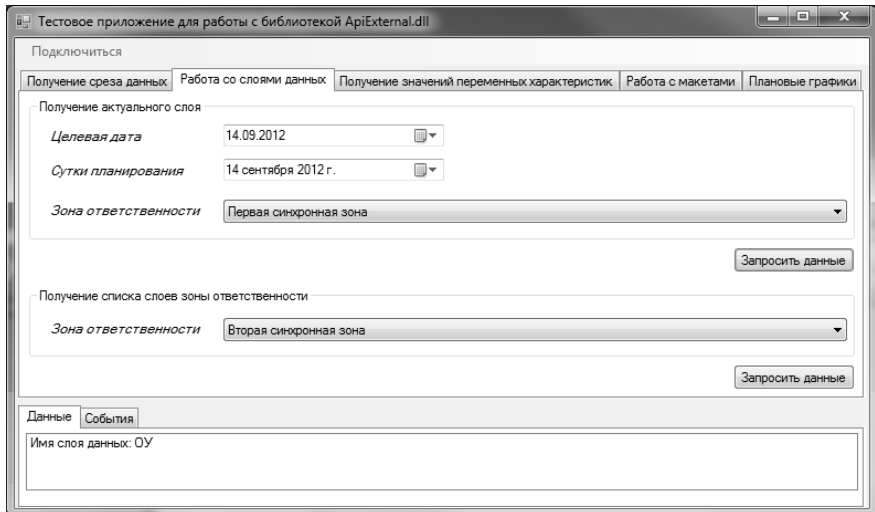


Рис. 3. Результат запроса актуального слоя

Результат запроса всех слоёв данных по второй синхронной зоне:

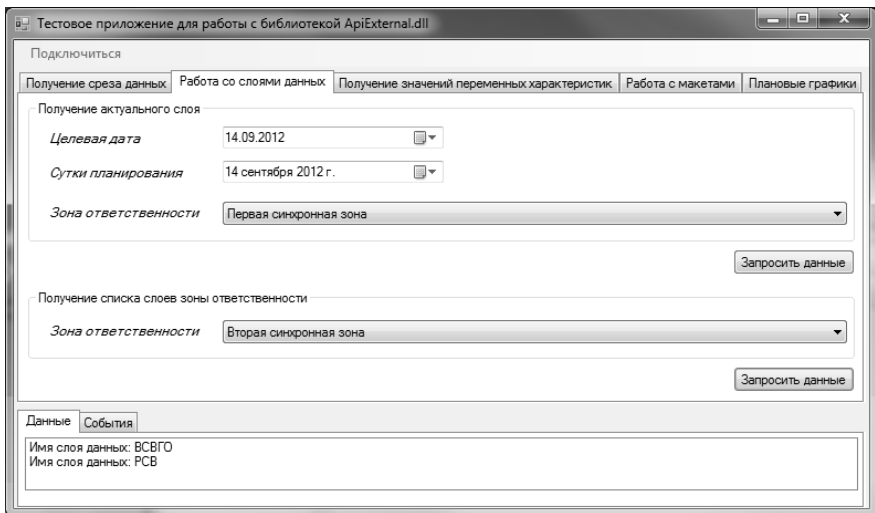


Рис. 4. Результат запроса всех слоёв данных по второй синхронной зоне

2.3. Получение значений переменных характеристик

На вкладке **Получение значений переменных характеристик** предоставляется возможность увидеть значения всех переменных характеристик по указанному объекту.

Предварительно необходимо запросить срез данных на указанную дату. После получения среза данных отображаются данные оборудования, действующего на момент указанной целевой даты, указанного типа оборудования и по указанной синхронной зоне. При этом стали доступны для изменения элементы поля для запроса переменных характеристик.

Имя	Наименование	Тип
GIDROOGK	ОАО "РусГидро"	sub
721630	"Волжская ГЭС" филиал ОАО "РусГидро"	sub
321630	Волжская ГЭС	station
3550	Волжская ГЭС	gou
GVOLGESV	Волжская ГЭС 500 (ГГ 7-12, ГГ 17-23), 200 (ГГ1-6, ГГ13-16)	gtpg
3112	Волжская ГЭС ГГ1-6,13-16	rge_ges

Рис. 5. Отображение результата запроса среза данных по генерирующему оборудованию

Далее необходимо указать имя типа объекта и имя объекта, которые можно ввести вручную, либо выбрать из списка при активации соответствующей опции. Также необходимо указать слой данных, в который были записаны данные переменных характеристик.

Результат запроса: Тип объекта – ЕГО-генератор, Имя объекта – 100320. Слой данных – ВСВГО, Источник данных – Станция.

Тестовое приложение для работы с библиотекой ApiExternal.dll

Подключиться

Получение среза данных | Работа со слоями данных | **Получение значений переменных характеристик** | Работа с макетами | Плановые графики

Получение среза данных по ген. оборудованию

Целевая дата: 20 июня 2013 г.

Зона ответственности: Первая синхронная зона

Запросить данные

Получение списка значений переменных характеристик

Тип объекта: gen_g ☒ выбрать из списка

Имя объекта: 100320 ☒ выбрать из списка

Слой данных: ВСВГО

Источник данных: ☐ СО ☒ Станция

Запросить данные

Данные | События

Характеристика	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Состояние_ЕГО	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Рмакс	110	110	110	110	110	110	110	110	110	110	110	110	110	110	110	110
Рмин	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Раз_max	35	35	35	35	35	35	35	35	35	35	35	35	35	35	35	35
Раз_min	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Рис. 6. Отображение результата запроса списка значений переменных характеристик

2.4. Работа с макетами

Вкладка **Работа с макетами** предназначена для отображения информации о содержимом выбранного макета. Для этого следует выбрать диапазон времени и нажать верхнюю кнопку **Запросить данные**. Внизу окна, на закладке **Данные**, появится информация о макете, а именно: Отправитель, Дата отправки, Макет на дату, Этап планирования и уникальный идентификатор макета:

Отправитель	Дата отправки	Макет на дату	Этап планирования	Uid
Голоднюк Евгений Пет...	07.09.2012 8:10:15	09.09.2012 0:00:00	ВСВГО	46c9471d-4f5c-48a0-97f5-0007e97a324d
Дежурный техник ГЩУ	03.09.2012 10:30:27	05.09.2012 0:00:00	ВСВГО	2ba7ebd1-1a9f-48ae-946d-0036202b4105
ОДУ Сибири	...	01.09.2012 0:00:00	ОУ	4dbc72e8-c055-469a-bdf0-00578dc3e7b5
Стусь Александр Алекс...	04.09.2012 15:16:43	06.09.2012 0:00:00	PCB	f0b6cd9e-643c-40c3-93a5-0057a1d0d315
Электромонтер ГЩУ эл...	06.09.2012 11:01:32	08.09.2012 0:00:00	PCB	c73528b3-4838-4a16-9882-0059b17f1568
Дежурный техник ГЩУ	05.09.2012 9:43:48	13.09.2012 0:00:00	ВСВГО	f6e49d78-7824-4006-8265-007c679fed0c

Рис. 7. Отображение информации о макетах за определённый период

После этого появится возможность выбрать макет по уникальному идентификатору из выпадающего списка. После выбора интересующего макета надо нажать вторую кнопку **Запросить данные**. Внизу окна, в закладке **Данные**, появится информация о содержимом выбранного макета.

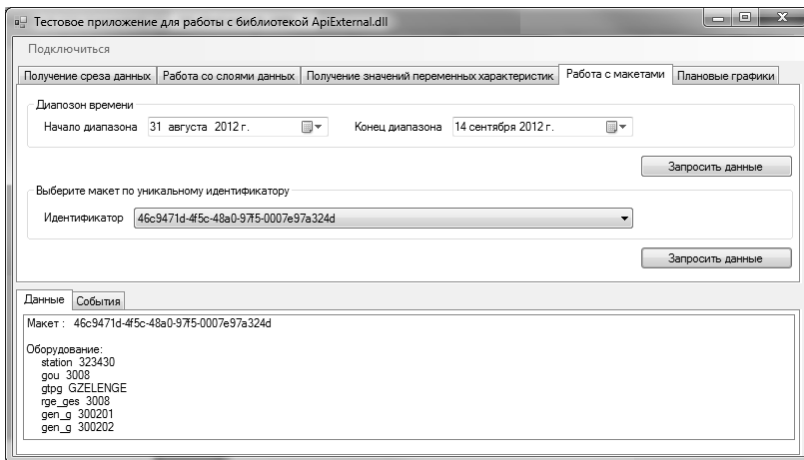


Рис. 8. Отображение информации о содержимом макета

2.5. Плановые графики

На вкладке **Плановые графики** предоставляется возможность увидеть значения плановых графиков. Для этого нужно выбрать **Зону ответственности** и выбрать оборудование.

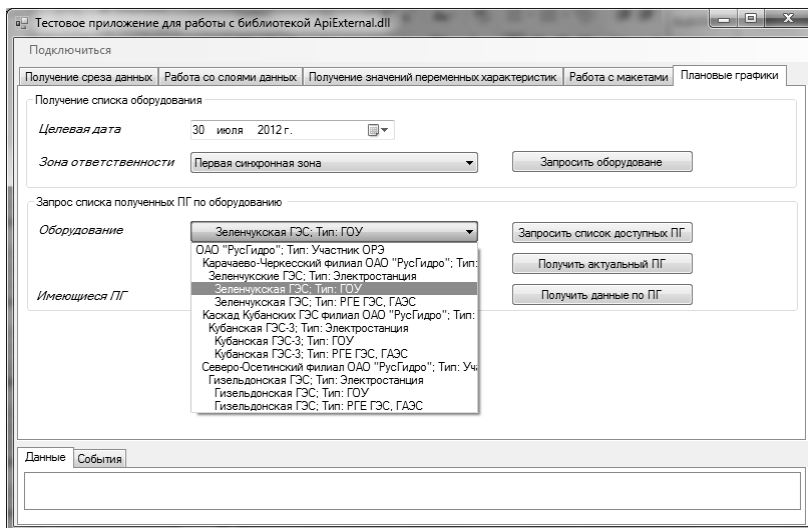


Рис. 9. Выбор оборудования для получения ПГ по нему

После чего пользователь может выполнить следующие действия:

- запрос актуальных значений Плановых графиков на выбранные сутки;
- запрос имеющихся плановых графиков по оборудованию на сутки;
- запрос значений определённого планового графика.

Характеристика	30 10:00	30 11:00	30 12:00	30 13:00	30 14:00	30 15:00	30 16:00	30 17:00
Генерация	140	140	140	140	140	140	140	140
Минимум генерации	0	0	0	0	0	0	0	0
Максимум генерации	140	140	140	140	140	140	140	140

Рис. 10. Результат запроса данных по ПГ

3. Редактор локальных правил

Клиентское приложение **Редактор локальных правил** предназначено для создания библиотек локальных правил автозаполнения – **AutoFillLocal.dll**, дорасчётов – **CalcLocal.dll** и проверок – **CheckLocal.dll** на языке программирования **C#**.

3.1. Интерфейс Редактора локальных правил

Интерфейс основного окна **Редактора локальных правил** представлен на следующем рисунке:

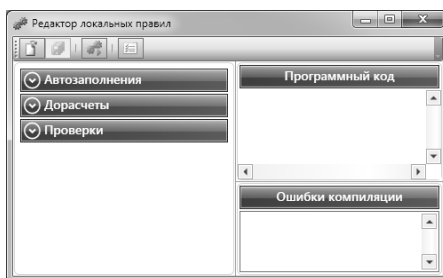






Рис. 11. Основное окно Редактора локальных правил

Окно редактора включает в себя области: выбора раздела редактируемой библиотеки, изменения программного кода и вывода ошибок компиляции; а также панель управления, состоящую из следующих инструментов:

-  – **Открыть** файлы библиотек автозаполнения, дорасчётов и проверок.
-  – **Сохранить всё** – сохранение всех изменений, внесённых в код в рамках сессии.
-  – **Построить правила** – запуск компиляции правил и создания библиотек.
-  – **Настройки** – вызов окна настроек, содержащего единственный параметр, в котором указывается путь каталога для сохранения файлов библиотек:

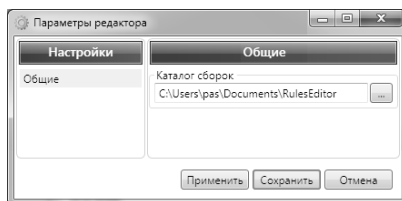


Рис. 12. Параметры редактора

3.2. Редактирование локальных библиотек

В состав дистрибутива включены локальные библиотеки: **AutoFillLocal.dll**, **CalcLocal.dll** и **CheckLocal.dll**, содержащие по одному примеру автозаполнения, дорасчётов и проверок. После загрузки данных библиотек имеется возможность создать собственные правила.

Для примера была создана при помощи **Редактора настроек** новая локальная характеристика – **КОЛ_вкл_бл** типа **Целочисленное** с привязкой к объекту **Электростанция**.

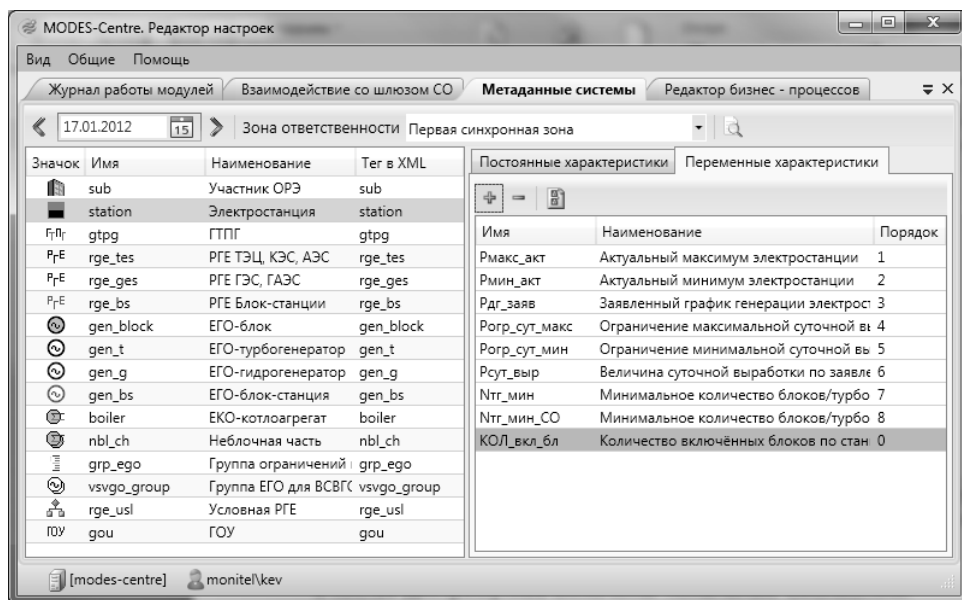


Рис. 13. Создание и привязка новой локальной характеристики

Параметры созданной характеристики были настроены следующим образом:

Редактирование связи с переменной характеристикой

Наименование связи	Начало	Окончание
Количество включённых блоков по станции на каждый интервал времени	01.01.2012	...

Зона ответственности

- ☒ Первая синхронная зона
- ☐ Вторая синхронная зона

Тип планирования

- ☒ Недельное планирование
- ☒ Суточное планирование
- ☐ Оперативное планирование

Наименование связи: Количество включённых блоков по станции на каждый интервал времени

Порядок отображения: 0

Значение по умолчанию: Нет

Тип связи: Вводно-дорасчетный

Шаг изменения: Часовой/Получасовой

Начало жизни: 01.01.2012

Окончание жизни: 27.01.2012

OK Отмена

Рис. 14. Параметры новой локальной характеристики



Подробное описание работы с **Редактором настроек** и в частности с вкладкой **Метаданные системы** приведено в разделе 5 Тома 1 данной документации.

Для данной переменной был подготовлен дорасчёт, записывающий значение в эту характеристику как сумму количества включённых блоков по станции на каждый интервал времени, и проверка на соответствие текущих значений характеристики этому правилу.

1. Подготовка дорасчёта:

В Редакторе локальных проверок необходимо открыть библиотеку **CalcLocal.dll** и на закладке **Дорасчёты** выбрать элемент **Электростанция**. В области **Программный код** надо внести необходимые изменения:

```
IVarParamsInt КОЛ_вкл_бл = genObj["КОЛ_вкл_бл"] as IVarParamsInt;
int КОЛ_вкл_бл_sum = 0;
if (КОЛ_вкл_бл != null)
{
    for (int i = 0; i < КОЛ_вкл_бл.PointCount; i++)
    {
        КОЛ_вкл_бл_sum = 0;
        if(genObj.Children!= null)
        if(genObj.Children.Count > 0)
        foreach (IGenObject obj1 in genObj.Children)
        {
            if (obj1.GenObjType.Name == "rge_tes" || obj1.GenObjType.Name
            == "rge_bs" || obj1.GenObjType.Name == "rge_ges")
            {
                if(obj1.Children!= null)
                if(obj1.Children.Count > 0)
                foreach (IGenObject obj in obj1.Children)
                {
                    if (obj.GenObjType.Name
                    == "gen_block" || obj.GenObjType.Name
                    == "gen_t" || obj.GenObjType.Name == "gen_g")
                    {
                        IVarParamsEnum Состояние_ЕГО
                        = (IVarParamsEnum)obj.GetVarParam("Состояние_ЕГО");
                        if (Состояние_ЕГО != null)
                        {
                            if(Состояние_ЕГО[i] != null)
                            {
                                string sost_val = Состояние_ЕГО.ValueStr[i];
                                if (sost_val == "Вкл")
                                    КОЛ_вкл_бл_sum += 1;
                            }
                        }
                    }
                }
            }
        }
        КОЛ_вкл_бл[i] = КОЛ_вкл_бл_sum;
    }
}
```

По завершению редактирования программного кода следует нажать кнопку **Построить правила:**

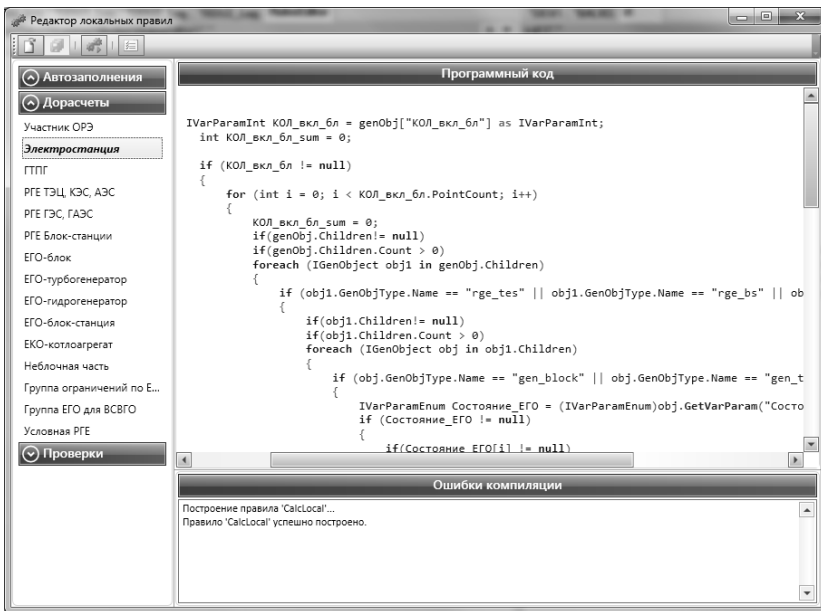


Рис. 15. Построение правил дорасчёта

В области **Ошибки компиляции** появятся сообщения об успешном построении правил. Файл **CalcLocal.dll** будет сохранён в каталог, указанный в настройках программы, по умолчанию:

C:\Users\<имя текущего пользователя>\Documents\RulesEditor

При наличии ошибки в правиле редактор выдаст соответствующее сообщение, и правило не будет построено.

2. Подготовка проверки:

В **Редакторе локальных проверок** необходимо открыть библиотеку **CheckLocal.dll** и на закладке **Проверки** выбрать элемент **Электростанция**. В области **Программный код** надо внести необходимые изменения:

```
IVarParamsInt КОЛ_вкл_бл = (IVarParamsInt)genObj["КОЛ_вкл_бл"];  
if (КОЛ_вкл_бл == null)
```

```
{
    Debug.WriteLine(String.Format("CheckLocal.dll: Для объекта {0}:{1}
    не найден условно-переменный параметр
    {2}", genObj.GenObjType.Name, genObj.Name, "КОЛ_вкл_бл"));
}
#region КОЛ_вкл_бл
int КОЛ_вкл_бл_sum = 0;
bool isRGETes = false;
if (КОЛ_вкл_бл != null)
{
    for (int i = 0; i < КОЛ_вкл_бл.PointCount; i++)
    {
        #region 100500
        if (КОЛ_вкл_бл[i] != null)
            if (КОЛ_вкл_бл[i] < 0)
                genObj.SetCheckResult(КОЛ_вкл_бл, i,
                String.Format("Значение КОЛ_вкл_бл Станции {0} не может
                быть меньше нуля", genObj.Description),
                CheckResultType.Error, 100500);
        #endregion
        КОЛ_вкл_бл_sum = 0;
        foreach (IGenObject obj1 in genObj.Children)
        {
            if (obj1.GenObjType.Name == "rge_tes" || obj1.GenObjType.Name
            == "rge_bs" || obj1.GenObjType.Name == "rge_ges")
            {
                isRGETes = true;
                foreach (IGenObject obj in obj1.Children)
                {
                    if (obj.GenObjType.Name
                    == "gen_block" || obj.GenObjType.Name
                    == "gen_t" || obj.GenObjType.Name == "gen_g")
                    {
                        IVarParamEnum Состояние_ЕГО =
                        obj.GetVarParam("Состояние_ЕГО") as IVarParamEnum;
```

```
if (Состояние_ЕГО != null)
{
    string sost_val = Состояние_ЕГО.ValueStr[i];
    if (sost_val == "Вкл")
    {
        КОЛ_вкл_бл_sum += 1;
    }
}
Else
{
    Debug.WriteLine(String.Format("CheckLocal.dll: Для
    объекта {0}:{1} не найден условно-постоянный
    параметр {2}", obj.GenObjType.Name, obj.Name,
    "Состояние_ЕГО"));
}
}
}
}
}
if (КОЛ_вкл_бл[i] != null)
{
    if (КОЛ_вкл_бл_sum < КОЛ_вкл_бл[i] && isRGEtes == true)
    {
        genObj.SetCheckResult(КОЛ_вкл_бл, i, String.Format("По
        Станции {0} указанное минимальное количество
        блоков/турбогенераторов, находящихся во включенном
        состоянии, превышает число ЕГО, находящихся в состоянии
        \"Вкл\"", genObj.Description), CheckResultType.Error, 6);
    }
}
}
}
}
#endregion
```

По завершению редактирования программного кода следует нажать кнопку **Построить правила:**

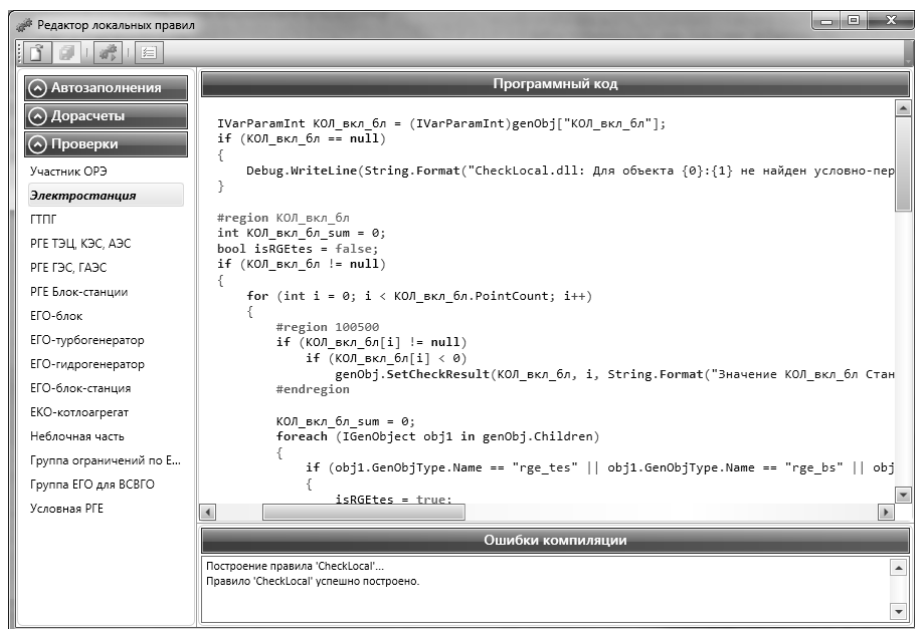


Рис. 16. Построение правил проверки

В области **Ошибки компиляции** появятся сообщения об успешном построении правил. Файл **CheckLocal.dll** будет сохранён в каталог, указанный в настройках программы по умолчанию:

C:\Users\<имя текущего пользователя>\Documents\RulesEditor

При наличии ошибки в правиле редактор выдаст соответствующее сообщение, и правило не будет построено.

3.3. Публикация локальных библиотек

После подготовки правил библиотеки необходимо опубликовать в **Редакторе настроек**. Для этого надо воспользоваться системным меню: **Общие** | **Локальные модули**, после чего появится следующее окно:

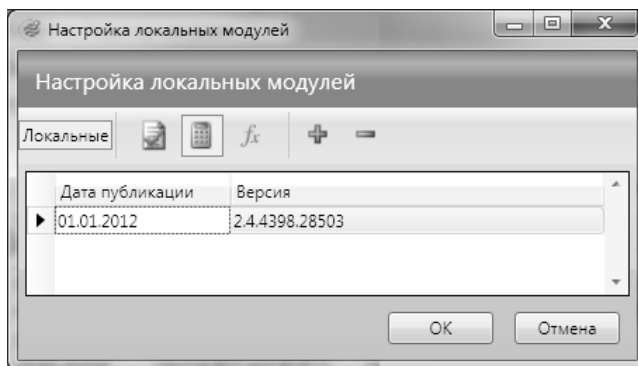


Рис. 17. Публикация модуля

Панель инструментов данного окна предоставляет следующий функционал:

Локальные / **Централизованные** – отображение информации по локальному или полученному централизованно модулю, соответственно.

/ / **fx** – отображение информации по библиотеке проверки, дорасчётов или автозаполнения соответственно.

+ / **-** – добавление/удаление модуля в/из системы.

После публикации библиотек следует запустить **Клиент MODES-Centre** и проверить в соответствии с описанным примером, что характеристика добавлена. Затем заполнить станцию и дочерние объекты значениями по умолчанию. Убедиться, что дорасчёт выполнен корректно. Также следует изменить значение характеристики **КОЛ_вкл_бл** на заведомо неверное и запустить проверку, чтобы увидеть, что в протоколе проверки появилась соответствующая ошибка, и она выделена красным цветом в окне данных.

4. Хранимые процедуры

4.1. Процедура GetPlanGraph

Процедура получения значений загруженных в систему плановых графиков.

Описание параметров:

- **@dtTarget** – сутки планирования в локальном времени СО.
Задаётся в формате ГГГГММДД чч:мм.
- **@name** – перечень идентификаторов оборудования, по которому требуется получить данные.
Значение столбца name таблицы [dbo].[MODES_Objects].
- **@objType** – перечень типов идентификаторов оборудования, по которому требуется получить данные.
Значение столбца id таблицы [dbo].[MODES_Objects].
- **@idArea** – синхронная зона.
0 – первая синхронная зона, 1 – вторая синхронная зона.
- **@pgType** – тип планового графика.
0 – ППБР, 1 – ПБР-01, 2 – ПБР-02 и т.д.

Пример использования:

```
EXECUTE [MODES].[dbo].[GetPlanGraph]
@dtTarget = '2012.06.14 21:00'
@name = '322630'
@objType = '1'
@idArea = '0'
@pgType = '0'
GO
```

4.2. Процедура GetPlanVersion

Процедура получения версий загруженных в систему плановых графиков.

Описание параметров:

- **@dtTarget** – сутки планирования в локальном времени СО.

Задаётся в формате ГГГГММДД чч:мм.

- **@name** – перечень идентификаторов оборудования, по которому требуется получить данные.

Значение столбца name таблицы [dbo].[MODES_Objects].

- **@objType** – перечень типов идентификаторов оборудования, по которому требуется получить данные.

Значение столбца id таблицы [dbo].[MODES_Objects].

Пример использования:

```
EXECUTE [MODES].[dbo].[GetPlanVersion]
@dtTarget = '2012.06.14 21:00'
@name = '322630'
@objType = '1'
GO
```

4.3. Процедура GetVarValuesByEqType

Процедура получения условно-переменных характеристик по типу генерирующего оборудования.

Описание параметров:

- **@dtStart** – стартовая дата, задаётся в локальном времени в формате: ГГГГММДД чч:мм.
- **@dtEnd** – конечная дата, задаётся в локальном времени в формате: ГГГГММДД чч:мм.
- **@syncZone** – синхронная зона.
0 – первая синхронная зона, 1 – вторая синхронная зона.
- **@idTask** – этап планирования.
0 – недельное планирование, 1 – суточное планирование, 2 – оперативное планирование.
- **@idSource** – источник данных.
1 – данные СО, 2 – данные станции.

- **@eqType** – перечень идентификаторов оборудования, по которому требуется получить данные.
Значение столбца name таблицы [dbo].[MODES_Objects].
- **@varParams** – идентификаторы условно-переменных характеристик.

Пример использования:

```
EXECUTE [MODES].[dbo].[GetVarValuesByEqType]
@dtStart = '2012.06.14 21:00'
@dtEnd = '2012.06.20 21:00'
@syncZone = '0'
@idTask = '0'
@idSource = '2'
@eqType = '3'
@varParams = 'Рмакс'
GO
```

4.4. Процедура GetVarValuesByEquipments

Процедура получения условно-переменных характеристик по генерирующему оборудованию.

Описание параметров:

- **@dtStart** – стартовая дата, задаётся в локальном времени в формате: ГГГГММДД чч:мм.
- **@dtEnd** – конечная дата, задаётся в локальном времени в формате: ГГГГММДД чч:мм.
- **@syncZone** – синхронная зона.
0 – первая синхронная зона, 1 – вторая синхронная зона.
- **@idTask** – этап планирования.
0 – недельное планирование, 1 – суточное планирование, 2 – оперативное планирование.
- **@idSource** – источник данных.
1 – данные СО, 2 – данные станции.

- **@equipments** – перечень идентификаторов оборудования, по которому требуется получить данные.
Параметр задаётся в виде ‘id оборудования, тип оборудования’.
- **@varParams** – идентификаторы условно-переменных характеристик.

Пример использования:

```
EXECUTE [MODES].[dbo].[GetVarValuesByEquipments]
    @dtStart = '2012.06.14 21:00'
    @dtEnd = '2012.06.20 21:00'
    @syncZone = '0'
    @idTask = '0'
    @idSource = '2'
    @equipments = '323033, station'
    @varParams = 'Рмакс'
GO
```

4.5. Процедура GetVarValuesByStation

Процедура получения условно-переменных характеристик по генерирующему оборудованию, находящемуся в ведении станции.

Описание параметров:

- **@dtStart** – стартовая дата, задаётся в локальном времени в формате: ГГГГММДД чч:мм.
- **@dtEnd** – конечная дата, задаётся в локальном времени в формате: ГГГГММДД чч:мм.
- **@syncZone** – синхронная зона.
0 – первая синхронная зона, 1 – вторая синхронная зона.
- **@idTask** – этап планирования.
0 – недельное планирование, 1 – суточное планирование, 2 – оперативное планирование.
- **@idSource** – источник данных.
1 – данные СО, 2 – данные станции.

- **@station** – перечень идентификаторов станций, по которым требуется получить данные.

Значение столбца name таблицы [dbo].[MODES_Objects].

- **@varParams** – идентификаторы условно-переменных характеристик.

Пример использования:

```
EXECUTE [MODES].[dbo].[GetVarValuesByStation]
```

```
@dtStart = '2012.06.14 21:00'
```

```
@dtEnd = '2012.06.20 21:00'
```

```
@syncZone = '0'
```

```
@idTask = '0'
```

```
@idSource = '2'
```

```
@station = '323033'
```

```
@varParams = 'Рмакс'
```

```
GO
```

4.6. Процедура GetVarValuesSlice

Процедура получения среза по генерирующему оборудованию.

Описание параметров:

- **@dtStart** – стартовая дата, задаётся в локальном времени в формате: ГГГГММДД чч:мм.
- **@dtEnd** – конечная дата, задаётся в локальном времени в формате: ГГГГММДД чч:мм.

- **@source** – источник данных.

1 – данные СО, 2 – данные станции, 5 – отправленные данные.

- **@task** – слой планирования.

0 – ВСВГО, 1 – РСВ, 2 – ОУ, 3 – актуальный на сутки запроса данных.

- **@step** – шаг дискретизации в секундах.

Для получения исходного шага требуется задать значение '-1'.

- **@spreadDayParam** – распространение суточных параметров.

0 – нет, 1 – да.

- **@timeType** – формат времени.

0 – сутки планирования, 1 – UTC, 2 – локальное время сервера ПАК MODES-Centre.

- **@equipments** – перечень внутренних идентификаторов оборудования, по которому требуется получить данные.

Значение столбца inner таблицы [dbo].[MODES_Objects].

- **@varParams** – идентификаторы условно-переменных характеристик.

Пример использования:

```
EXECUTE [MODES].[dbo].[GetVarValuesByStation]
```

```
@dtStart = '2012.06.14 21:00'
```

```
@dtEnd = '2012.06.20 21:00'
```

```
@task = '0'
```

```
@source = '2'
```

```
@step = '-1'
```

```
@timeType = '2'
```

```
@equipments = '167'
```

```
@varParams = '54'
```

```
GO
```

Перечень иллюстраций

<i>Рис. 1. Главное окно тестового приложения.....</i>	<i>23</i>
<i>Рис. 2. Отображение результата запроса среза данных.....</i>	<i>24</i>
<i>Рис. 3. Результат запроса актуального слоя.....</i>	<i>25</i>
<i>Рис. 4. Результат запроса всех слоёв данных по второй синхронной зоне</i>	<i>25</i>
<i>Рис. 5. Отображение результата запроса среза данных по генерирующему оборудованию.....</i>	<i>26</i>
<i>Рис. 6. Отображение результата запроса списка значений переменных характеристик.....</i>	<i>27</i>
<i>Рис. 7. Отображение информации о макетах за определённый период</i>	<i>28</i>
<i>Рис. 8. Отображение информации о содержимом макета</i>	<i>29</i>
<i>Рис. 9. Выбор оборудования для получения ПГ по нему.....</i>	<i>29</i>
<i>Рис. 10. Результат запроса данных по ПГ.....</i>	<i>30</i>
<i>Рис. 11. Основное окно Редактора локальных правил</i>	<i>31</i>
<i>Рис. 12. Параметры редактора.....</i>	<i>31</i>
<i>Рис. 13. Создание и привязка новой локальной характеристики</i>	<i>32</i>
<i>Рис. 14. Параметры новой локальной характеристики</i>	<i>33</i>
<i>Рис. 15. Построение правил дорасчёта.....</i>	<i>35</i>
<i>Рис. 16. Построение правил проверки.....</i>	<i>38</i>
<i>Рис. 17. Публикация модуля.....</i>	<i>39</i>

Условия использования

Настоящий документ пересматривается на регулярной основе с внесением всех необходимых исправлений и дополнений в следующие выпуски.

Предприняты все меры для того, чтобы содержащаяся здесь информация была максимально актуальной и точной, тем не менее, компания Монитор Электрик не несёт ответственности за ошибки или упущения, а также за любой ущерб, причинённый в результате использования содержащейся здесь информации.

О технических неточностях или опечатках вы можете сообщить по адресу электронной почты support-modes@monitel.ru. Мы будем рады вашим замечаниям и предложениям.

Содержание данного документа может быть изменено без предварительного уведомления. Перед использованием убедитесь, что это актуальная версия, соответствующая версии используемой системы. Для получения актуальной версии вы можете обратиться по адресам, указанным на сайте www.monitel.ru.

Данный документ содержит информацию, которая является конфиденциальной и принадлежит Монитор Электрик. Все права защищены. Не допускается копирование, передача, распространение и иное разглашение содержания данного документа, а также, любых выдержек из него третьим лицам без письменного разрешения Монитор Электрик. Нарушители несут ответственность за ущерб в соответствии с законом.

Названия продуктов и компаний, упомянутые здесь, могут являться торговыми марками соответствующих владельцев.