Министерство образования Республики Беларусь
Учреждение образования
“Белорусский государственный университет
информатики и радиоэлектроники”

Факультет информационных технологий и управления
Кафедра интеллектуальных информационных технологий

К защите допустить
Заведующий кафедрой ИИТ
_____ В.В. Голенков
«___» _____ 2009 г.

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к расчетной работе
по дисциплине «ПИСИСПР»
на тему
«Создание административной утилиты
управления серверными приложениями по протоколу FIX»
БГУИР КП8 1 – 40 03 01 02 56 ПЗ

Выполнил ( )
студент группы
521703 Сидоров И. С.
Руководитель ( )
Степанова М. Д.

МИНСК
2009

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Для того, чтобы увеличить количество потенциальных участников рынка ценных бумаг была создана возможность заключения сделок по торговле ценными бумагами, использующая средства компьютерной техники. Для этого было разработано множество различных средств и приложений. Для управления такими приложениями требуются дополнительные приложения - утилиты. Утилиты используются для контроля работы систем предоставляющих доступ к электронной торговле.

Для этих целей был разработан протокол передачи информации под названием «FIX» (электронный протокол передачи информации о финансах). Данный протокол является независимым от методов передачи информации (сетевое взаимодействие, использование файлов проецируемых в память и др.).

Протокол включает в себя набор администраторских сообщений для управления, и анализа работы приложений серверов предоставляющих доступ к электронным рынкам.

Цель работы: создание приложения которое может: изменять настройки серверов «FIX», управлять подключёнными клиентами к серверу «FIX», работать с несколькими серверами одновременно. Задачами становятся:

- изучить протокол взаимодействия «FIX»;
- проанализировать существующие приложения управления;
- создать приложение, реализующее функции по управлению серверных приложений «FIX»;
- создать руководство пользователя, с помощью которого пользователь сможет настраивать приложение для своих целей.

1. АНАЛИЗ АДМИНИСТРАТИВНОЙ ЧАСТИ ВОЗМОЖНОСТЕЙ ПРОТОКОЛА «FIX»

1.1. Назначение и описание протокола «FIX», описание административных команд.

Для улучшения процесса торговли в мировом масштабе в задачах: определения, управления и информирования был разработан открытый протокол электронного взаимодействия между участниками отрасли.

Данный протокол является международным стандартом для обмена данными между участниками биржевых торгов и режиме реального времени. Изначально протокол разработан в 1992 году для передачи информации о торгах акциями между компаниями «Fidelity Investments» и «Salomon Brothers». В настоящее время широко используется торговыми системами для обмена финансовыми данными и совершения транзакций.

Протокол «FIX» поддерживается большинством крупнейших банков и электронными торговыми системами, а также биржами мира.

Данный протокол является само описывающимся. «FIX» протокол — протокол сессионного уровня. Сессия «FIX» проводится поверх «TCP» (Transmission Control Protocol).

Сообщения «FIX» состоят из набора полей, причём каждое поле — это пара «тег-значение». Поля отделяются друг от друга специальным кодом «SOH» (Start of Header (0x01)). Тег — это строковое представление целого числа, которое обозначает номер поля. Значение — это последовательность байтов. Например, тег «48» обозначает «securityID», а строка значения означает идентификатор ценной бумаги.

Протокол «FIX» определяет ряд обязательных полей сообщения,

остальные поля являются необязательными. В общем случае последовательность полей неважна. Сообщение делят на три части «заголовок», «тело сообщения», «окончание». В качестве последнего поля обязательно должны находиться данные с контрольной суммой (тег 10).

На данный момент наиболее широко используются две версии протокола «FIX» 4.2, 4.4. Версия протокола передаются в заголовке каждого «FIX» сообщения в первом теге (8).

Расширяемость протокола, позволяет использовать его как в направлении предназначения, так и собственного использования. «FIX» («The Financial Informational eXchange» - протокол обмена финансовой информацией) протокол был разработан для электронного взаимодействия и пересылки сообщений, связанных с электронной коммерцией. В разработке протокола участвовали банки, брокеры-дилеры, участники торгов, инвесторы и множество системных аналитиков различных государств. Успех использования протокола «FIX» в финансовом сообществе было подтверждено эмпирическим путём, что доказано в результате полномасштабного исследования проведённого «TowerGroup». На данный момент 75 процентов финансовых сделок совершается при помощи протокола «FIX».

Протокол предоставляет широкие возможности администрирования торговых и информационных площадок, предоставляющих доступ к различным рынкам. Администрирование «FIX» серверов происходит двумя способами:

1) изменение файлов настройки «FIX» серверов (такие настройки применяются только после перезапуска сервера);

2) изменение настроек на лету предоставляемые протоколом «FIX», такие настройки изменяются в момент их исправления, но не сохраняются после перезапуска.

Таким образом, администрировать приложение требуется двумя различными способами:

1) изменение файла настроек с помощью дополнительного файлового сервера (FTP);
2) подключение к серверу «FIX», и отправка «FIX» сообщений серверу.

Административные возможности предоставляемые протоколом «FIX» можно описать следующим списком функций:

1) «среднестатистическое количество полученных данных»;
2) «среднестатистическое количество отправленных данных»;
3) «среднестатистическое количество корректных данных»;
4) «изменить нумерацию передаваемых сообщений для определённого клиента»;
5) «создать возможность подключения для нового клиента»;
6) «отключить клиента»;
7) «отключить всех клиентов»;
8) «информация о сессиях клиентов»;
9) «проверить определённого клиента на то, что он всё ещё подключён»;
10) «суммарная статистика по обработанным сообщениям»;
11) «суммарная статистика по полученным данным»;
12) «обнулить нумерацию передаваемых сообщений для определённого клиента»;
13) «отправить сообщение определённому клиенту»;
14) «суммарная статистика по отправленным данным»;

15) «получить информацию о статусе определённого клиента»;

16) «получить список сессий с информацией о подключении»;

17) «получить статистическую информацию по сессии»;

18) «отправить тестовое сообщение клиенту, предупреждающее о скором возможном отключении»;

19) «сохранить информацию об определённой сессии в файл»;

20) «отправить сообщение 'e-mail' списку сессий»;

21) «получить информацию об определённой сессии»;

Данные административные команды позволяют разработать приложение, отображающее сессии в виде списка, и проделывающее требуемые операции с выбранными сессиями.

Требуется добавить, что для корректной работы с протоколом «FIX» можно использовать библиотеку «B2BITS FIX-Antenna», а для работы с «XML» сообщениями библиотеку «Xerces-C».

1.2. Анализ существующих приложений администрирующих сервера «FIX».

На данный момент существует только одно приложение для управления серверами «FIX». Это приложение «FAMonitorTool». Данное приложение на вход принимает путь к файлу хранящему «XML» сообщение, характеризующее вызываемую административную функцию.

Взаимодействие пользователя с данным приложением представляет собой, написание «XML» файла для каждой команды, при этом приложение для отправки сообщения серверу подключается заново, после чего отправляет выбранное пользователем сообщение, получает результат и выводит его на экран.

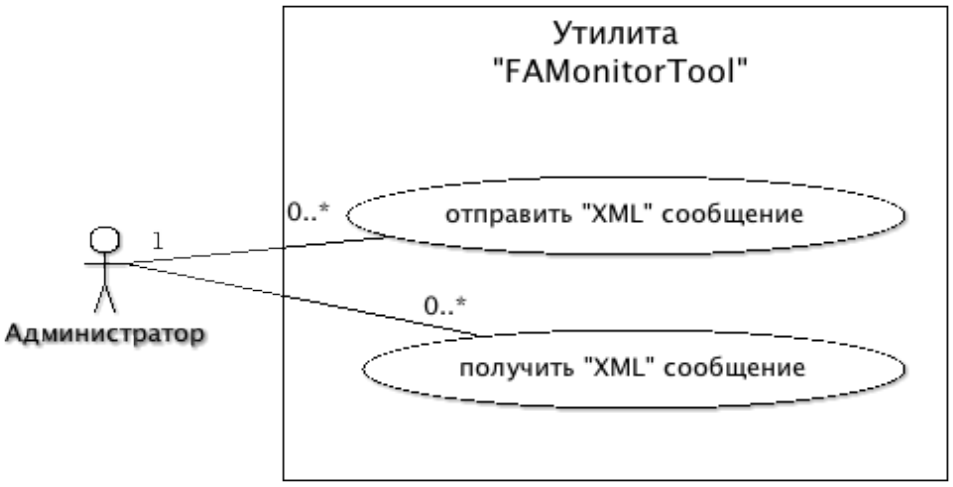


Рисунок 1.1: Диаграмма вариантов использования утилиты "FAMonitorTool"

Данная диаграмма описывает взаимодействие пользователя с приложением, для управления «FIX» сервером. Диаграмма позволяет увидеть, что приложение позволяет отправлять команды на сервер только, в виде готовых «XML» сообщений. Это означает, что пользователь должен вручную формировать сообщения. Способ вывода на экран в виде текстового представления «XML» сообщения не является удобным для конечного пользователя. Особенно это касается сообщений содержащих список некоторых объектов.

Для каждой определённой функции, которая возвращает отдельное «XML» сообщение (или группу сходных «XML» сообщений), приложение должно предоставлять отдельный способ предоставления ответа.

Например, для вывода списка сессий удобно использовать таблицу отображающую: колонки с данными о сессии, позволяющими определить к какому пользователю относится данная сессия и колонку отображающую состояние сессии (подключена, отключена, ожидает подключения).

Таким образом, нашей задачей становится написание административной утилиты «MFBM», позволяющей выполнять часть административных функций таким образом, чтобы пользователь мог выполнять быстро такие функции, и удобно получать информацию о результате выполнения этих функций.

Таблица 1.1. Сравнение возможностей существующей и разрабатываемой утилит.

Название возможности	«FAMonitorTool»	«MFBM»
Отправка и получение абсолютно любого административного сообщения	+	-
Поддержка подключения и связная отправка сообщений.	-	+
Кросс - платформеность	+	-
Графический интерфейс	-	+
Поддержка нескольких серверов одновременно.	-	+
Автоматическое оповещение пользователя о том невозможности доступа к серверу.	-	+
Возможность подключения к серверу «FTP» для чтения-записи файла настроек изменяемого приложением.	-	+
Возможность настройки графических диалоговых окон, для возможности изменения графической составляющей административной утилиты.	-	+

1.3. Требования к административной утилите управления программными серверами «FIX».

Определим список требований к разрабатываемой административной утилите «MFBM» управления программными серверами «FIX»:

- графический интерфейс;
- поддержка нескольких «FIX» серверов одновременно;
- отображение списка сессий вместе их статусами в качестве основной информации о каждом «FIX» сервере;
- возможность информирования группы клиентов по средствам отправки 'e-mail' сообщений;
- возможность отключения выбранных клиентов от сервера;
- возможность информирования пользователя «MFBM» о потере соединения с сервером;
- возможность предварительной настройки набора серверов которые требуется подключить;
- возможность, специфицированния определённых диалоговых окон и меню приложения в файле настройки;
- возможность подключения к определённому в настройках «FTP» серверу, для получения файла настроек сервера, изменения его на лету и сохранения обратно на «FTP» сервер.

Данный набор требований, позволит создать удобное для конечного пользователя приложение. Такое приложение позволит выполнять определенный набор административных возможностей, позволяющих быстро и

качественно администрировать несколькими серверами «FIX» одновременно.

Основной упор делается на удобство взаимодействия пользователя с приложением, а также возможности изменения множества настроек приложения без изменения исходных кодов и повторной сборки приложения.

2. ПРОЕКТИРОВАНИЕ АДМИНИСТРАТИВНОЙ УТИЛИТЫ УПРАВЛЕНИЯ ПРОГРАММНЫМИ СЕРВЕРАМИ «FIX»

2.1. Задачи, решаемые административной утилитой.

Для успешного проектирования и последующей разработки требуется определить набор задач решаемых административной утилитой управления программными серверами «FIX». Выделим набор таких задач и шагов к их решению для последующего проектирования:

1. Задача наблюдения за состоянием работы нескольких серверов. Для решения этой задачи административная утилита должна постоянно поддерживать соединение с выбранными серверами, при потере соединения требуется оповещать пользователя.
2. Задача просмотра списка сессий одного из серверов и состояния такого подключения. Для решения этой задачи административная утилита должна запрашивать состояние подключений у сервера с некоторой периодичностью. При этом выводить результаты в табличном виде, организуя для каждого сервера отдельную таблицу. При получении новых данных обновление текущей просматриваемой таблицы, не должно перелистывать список сессий вверх, а отображать то место списка, которое просматривает пользователь.
3. Задача подключения и отключения новых серверов во время работы утилиты. Для решения данной задачи, требуется сделать два диалоговых окна, одно в котором заполняются данные по подключению. А во втором выбирается подключённый сервер, для отключения. Диалоговые окна могут быть открыты, соответствующим выбором в главном меню приложения.
4. Задача подключения заранее определённого списка серверов. Для

решения данной задачи административная утилита должна использовать дополнительный файл настроек, в котором некоторым образом будут определяться предварительные настройки подключения. Для вызова набора подключений к набору серверов в главном меню приложения требуется создать кнопку.

5. Задача отправки специального сообщения ‘e-mail’ выбранному набору клиентов. Для этого в административной утилите таблица отображающая список сессий сервера должна получить следующие возможности:

    1. Выделение нескольких серверов (выделение происходит всех строки таблица характеризующей отдельную сессию);
    2. Вызов контекстного меню, в котором существует выбор «Отправить сообщение»;

    Для введения необходимых параметров сообщения, требуется использовать дополнительное диалоговое окно.

6. Задача отключения от сервера «FIX» выбранного пользователем клиента. Для этого в административной утилите контекстное меню таблицы отображающей список сессий получает дополнительную кнопку «Удалить», отключающую выбранную сессию на сервере к которому относится данная таблица.

7. Задача возможности экспорта данных из просматриваемой таблицы в текстовый файл. Для реализации этой возможности административная утилита должна быть дополнена следующим набором элементов управления:

    1. Дополнительная кнопка главного меню, вызывающее диалоговое окно сохранения нового файла экспортирующая всю

просматриваемую таблицу в текстовый файл.

2. Дополнительный пункт контекстного меню в таблицах, позволяющий экспортировать выбранные строки.

8. Задача получения файла настроек с сервера, его изменения на лету, и сохранения обратно на сервер, используя протокол передачи файлов “FTP”. Для решения данной задачи определим следующие правила действия:

    1. пользователь должен знать путь к редактируемому серверу и оформить доступ к файлу настроек сервера в файле настроек администраторской утилиты «MFBM»;

    2. главное меню должно быть дополнено кнопками загрузки файла с сервера, и загрузки такого файла обратно на сервер после редактирования;

    3. список подключённых серверов дополняется двумя таблицами настроек сервера. Первая таблица «Сервера» содержит настройки подключения сервера к рыночным данным, вторая таблица «Сессии» содержит настройки подключения клиентов к серверу;

    4. настройки содержимого таблиц должны быть динамическими, для возможности подстройки приложения под конкретные сервера, в которых такие настройки подключения могут быть специфическими. Для этого требуется организовать систему динамических таблиц для вывода таких настроек в файле настроек административной утилиты «MFBM».

Таким образом, мы получили список задач, которые должна решать проектируемая утилита администрирования. Данные задачи и примерные

описания их решений помогут нам правильно спроектировать приложение, и составить наборы действий пользователя для выполнения той или иной задачи.

2.2. Проектирование вариантов использования административной утилиты.

Опишем варианты использования административной утилиты «MFDB» управления серверами «FIX». Для этого предоставим следующую диаграмму.

Рисунок 2.1: Диаграмма вариантов использования административной утилиты "MFBM".

Данная диаграмма полностью описывает варианты использования утилиты администрирования.

Опишем действия пользователя для выполнения каждого из возможных вариантов:

1. «настраивать систему администрирования»: для этого пользователь должен отредактировать файл настроек ‘MFBM.ini’;
    1. «редактировать файл настроек ‘MFBM.ini’»: для этого

пользователь может воспользоваться любым текстовым редактором, поставляемым вместе с операционной системой (‘notepad’);

2. «выбирать подключённый сервер»: пользователь для осуществления такого выбора, должен выбрать идентификатор требуемого сервера в списке серверов;
    1. «просматривать список сессий»: после осуществления выбора конкретного сервера в списке серверов, перед пользователем должна отобразиться таблица содержащая список сессий для данного «FIX» сервера;
        1. «экспортировать выбранные данные в файл»: пользователь может выбирать данные посредствам возможности множественного выбора в отображаемых таблицах, экспорт происходит при выборе соответствующего действия в контекстном меню, или выборе действия «Экспортировать» в главном меню;
3. «отключать клиентов от сервера»: соответствующее действие должно быть доступно в контекстном меню при выборе конкретного пользователя в списке сессий сервера «FIX»;
4. «отправлять сообщение клиентам сервера»: данное действие должно быть доступно в соответствующем меню при выборе некоторого набора сессий среди существующих для выбранного сервера «FIX», пользователю должна быть доступна возможность вводить как само сообщение (некоторый упорядоченный список строк), так и тему сообщения (строковое значение);
5. «получать файл настроек сервера по ‘FTP’»: соответствующая

возможность должна быть предоставлена при выборе пользователем соответствующего пункта главного меню, при этом приложение будет подключаться по ‘FTP’ к заранее настроенному серверу, получать его файл настроек, обработать и генерировать специализированные таблицы настроек «Сервера», «Сессии»;

6. «изменять файл настроек сервера»: данная возможность предоставляется при помощи динамический таблиц, и набора динамических диалогов – настраиваемых в файле настроек «MFBM.ini»;
7. «отправлять изменения по FTP»: данный вариант, должен использовать существующее соединение (или создавать новое соединение к ‘FTP’ серверу) и сохранять на ‘FTP’ сервер изменённый файл настроек;

Спроектировав варианты использования системы администрирования «FIX» серверов, мы специфицировали как возможности пользователя работающего с системой, так и способы доступа к этим возможностям. Также диаграмма вариантов использования показывает, какие, в какой очерёдности требуется проектировать – реализовывать будущие возможности проектируемой системы.

Опишем порядок реализации, и последующего тестирования таких возможностей:

1. Требуется специфицировать-реализовать возможность «настраивать систему администрирования». Для этого требуется, специфицировать название файла настроек («MFBM.ini»), и при запуске приложения проверять его на корректность, считывать.

   Данную операцию мы будем проектировать в разделе проектирования пользовательского интерфейса.

2. Требуется специфицировать пользовательский интерфейс выбора одного из подключённых «FIX» серверов. А также пользовательский интерфейс отображения списка сессий для выбранного пользователем «FIX» сервера.

   Данную операцию мы будем проектировать в разделе проектирования пользовательского интерфейса.

3. Требуется специфицировать возможность подключения, получения данных с «FIX» серверов, а также отключения от уже подключённых серверов «FIX».

   Данную возможность мы будем проектировать в разделах проектирования модуля работы с «XML» сообщениями, проектирование модуля общения с программными серверами «FIX». Т.к. “FIX” протокол специфицировал группу команд передаваемых от административных приложений к серверам таким образом, что для передачи команды требуется передавать «XML» сообщение внутри «FIX» сообщений.

4. Требуется специфицировать возможность «отключать клиентов от сервера», данная возможность уже может быть спроектирована после выполнения предыдущих операций.

5. Требуется специфицировать возможность «отправлять сообщение клиентам сервера». Данная возможность может быть спроектирована в последующих разделах при специфицировании предыдущих возможностей.

6. На этом шаге требуется специфицировать задачи связанные с получением данных файла настроек по ‘FTP’, редактированием такого файла. И отправкой изменённого файла обратно.

7. На данном шаге требуется специфицировать возможность сохранения отображаемых данных в текстовый файл (которая называется «экспортировать выбранные данные в файл»).

   Данная возможность является последней возможностью, потому как она зависит от множества предыдущих возможностей. Её проектирование будет совершенно в подразделе проектирования взаимодействия между модулями.

2.3. Проектирование пользовательского интерфейса.

После определения всех необходимых для системы возможностей можно приступить к проектированию методов удобного доступа к этим возможностям. В качестве пользовательского интерфейса, мы будем использовать оконный пользовательский интерфейс, обычно включающий в себя, главное окно – и дополнительные диалоговые или информационные окна.

Главное окно, должно содержать:

А) список подключённых серверов «FIX»;

Б) список из двух дополнительных таблиц, которые могут быть добавлены к первому списку;

В) главное меню, для выполнения дополнительных операций;

Г) область отображения как данных о сессиях, так и данных файла настроек (двух таблиц «Сессии», «Серверы»);

Изобразим схему главного окна административной утилиты «MFBM» управляющей серверами «FIX».

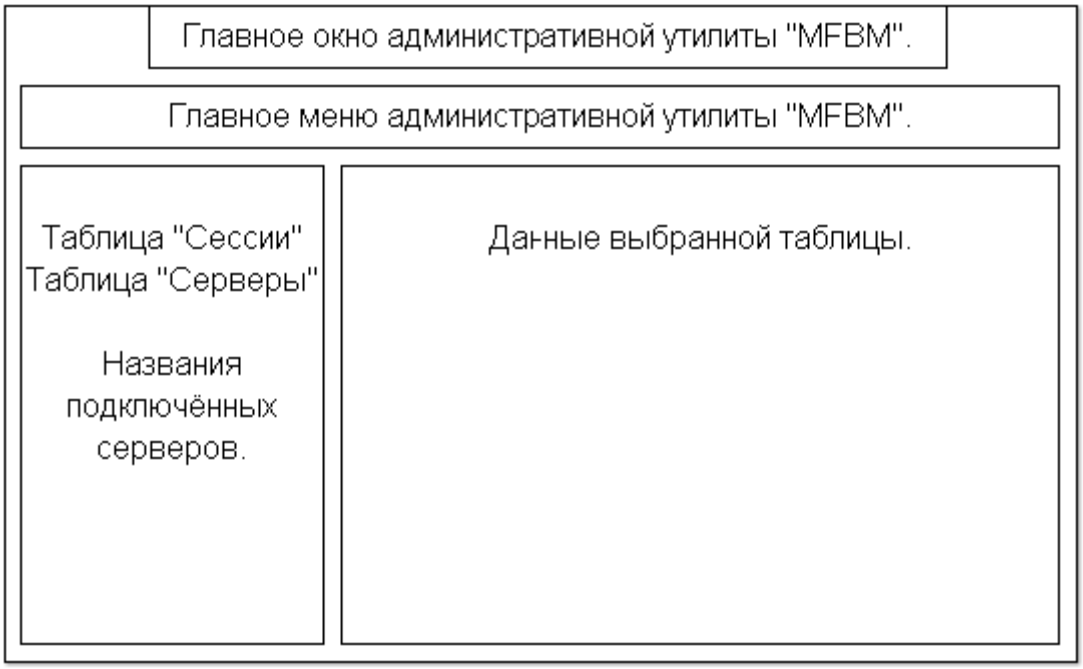


Рисунок 2.2. Схема главного окна утилиты "MFBM"

Таким образом, мы определили, как должно выглядеть главное окно административной утилиты. Данный вариант главного окна позволяет удобно и быстро переключаться между просматриваемыми серверами. А также наблюдать за реакцией какого-либо одного сервера, управляя им, с помощью определённого нами списка команд.

Представим список возможностей доступ, к которым необходим через главное меню административной утилиты “MFBM”:

1) возможность подключения к новому серверу «FIX»;
2) возможность отключения от подключённого сервера «FIX»;
3) возможность подключения ко всем заранее сконфигурированным серверам «FIX»;
4) возможность закрытия административной утилиты;
5) возможность загрузки файла настроек заранее сконфигурированного сервера «FIX» по протоколу «FTP»;

6) возможность сохранения на “FTP” сервер ранее загруженного файла настроек сервера «FIX»;
7) возможность экспортирования видимых данных в файл на жестком диске.

Данные возможности можно группировать следующим образом:

1. «Файл»:
    1. «Загрузить»;
    2. «Сохранить»;
    3. «Экспортировать в файл»;
    4. «Выйти»;
2. «Работа с серверами»:
    1. «Подключиться»;
    2. «Отключится»;
    3. «Подключится со всем заранее настроенным».

Описав, таким образом, главное меню, мы можем точно: сказать какие возможности программы будут доступны пользователю в любой момент работы с приложением.

Разберемся с настройками, которые требуется хранить в файле настроек административной утилиты «MFBM». Такие настройки должны иметь возможность объединения в группы, позволяющие как то объединять и показывать взаимосвязь между группами настроек пользователю

Для этого определим следующие правила:

А) начало группы символизирует обозначение группы (название) в символах квадратных кавычек;

Б) начало следующей группы символизирует об окончании предыдущей.

Опишем группы настроек, и необходимые для хранения в них настройки:

1. Настройки подключения к ‘FTP’ серверу. Данные настройки требуют ввода некоторых параметров, для успешного подключения и общения с ‘FTP’ сервером. Среди них:
    1. ‘ip’ адрес сервера;
    2. Наименование пользователя ‘FTP’;
    3. Пароль пользователя ‘FTP’;
    4. Полный путь к файлу настроек сервера, на удалённом компьютере;
    5. Специализированный символ окончания файлов (обычно равен «LF»).
2. Настройки стандартного подключения к «FIX» серверу (настройки по умолчанию):
    1. Идентификатор администраторской сессии («SenderCompId»);
    2. Идентификатор наименования сервера для администраторской сессии («TargetCompId»);
    3. Кодовый идентификатор наименования администраторской сессии («Username»);
    4. Кодовый идентификатор секретного слова администраторской сессии («Password»);
    5. ‘ip’ адрес машины сервера «FIX», куда следует подключаться;
    6. ‘port’ машины сервера «FIX», куда следует подключаться;
    7. Время, через которое можно считать подключение потерянным;

3. Список наименований серверов, которые настраиваются предварительно и должны начать подключение по нажатию кнопки «Подключится со всем заранее настроенным» главного меню;

    1. Список наименований серверов через символ запятой.

4. Набор блоков содержащих конкретизированные параметры подключения (не конкретизированные параметры будут установлены из настроек подключения по умолчанию).

5. Настройка динамического графического интерфейса:

    1. Определение возможных типов;

        1. настройки сервера могут содержать тип, представляющий собой выбор одного из возможных типов подключения (перечисление из нескольких элементов), для такого параметра удобно использовать выпадающее окно выбора;
        2. настройки сервера могут содержать динамические типы списков (например, имена используемых серверов требуются для определения прав доступа возможных клиентских сессий).

    2. Определение графического представления:

        1. Настройки, специфицирующие данные, хранимые в файле настроек «FIX» сервера, такие настройки должны позволять изменять как отображаемые таблицы сервера, так и группу диалогов по созданию новых элементов, и редактированию существующих элементов в таблицах «Сессии» и «Серверы».

            Для определения взаимосвязанных настроек будем использовать специальные классификаторы,

разделяющиеся от названия двоеточием.

Данный вид классификаций, позволит нам, рекурсивно вызывая процедуру изображения, составлять все необходимые графические представления.

Спроектируем общую структуру файла настроек «MFBM.ini».

Таблица 2.1. Структура файла настроек «MFBM.ini».

№	Наименование настройки	Пример использования	Зависит от настройки номер
1	Настройка доступа к «FTP» файлу	[FTP] Address = 127.0.0.1 User = user Password = pass File = / MCX-Fix.properties EOL = LF	-
2	Настройка параметров подключения по умолчанию.	[MFDB_DEFAULT] SenderCompId = MonitoringTool1 TargetCompId = FIXADMIN Username = MonitoringTool Password = MonitoringToolPassword Host = 10.6.3.35 Port = 9111 Timeout = 5000	-
3	Список серверов для подключения по умолчанию	[MFDBS] Servers = localhost	4
4	Настройка конкретного сервера подключения по умолчанию	[localhost] SenderCompId = MonitoringToolLocal TargetCompId = FIXADMIN Host = 127.0.0.1	2
5	Настройки вкладок в списке серверов “FIX”	[UI] Pages = Properties, FIX CSV = ;	-

№	Наименование настройки	Пример использования	Зависит от
6	Настройка используемых типов при отображении данных	[Types] Lists = Permissions Enums = ServerInterface,Boolean	-
7	Настройка динамического списка состоящего из элементов, полученных из файла настроек после его прочтения.	[List:Permissions] Type = Dynamic Source = Column:UI\Properties\Servers\Table\Name	6
8	Настройка типа перечисления, из известных элементов.	[Enum:ServerInterface] Type = Static Values = IFC_INFO_03,IFC_INFO12	6
9	Настройки отображающей набор текстовых полей и других управляющих элементов.	[Page:UI\Properties] Pages = Servers,Sessions Type = Controls Controls = PID,Engine config.	5
10	Настройка управляющего элемента.	[Control:UI\Properties\PID] Type = String Source = MicexAdaptor.PID.FileName	9
11	Настройка, отображающая группу элементов («Сервера»).	[Page:UI\Properties\Servers] Type = Table	9
12	Настройка, отображающая таблицу из двух колонок («Сервера»).	[Control:UI\Properties\Servers\Table] Source = MicexAdapter.Servers Rows = MicexAdapter.Servers.Names Columns = Name,Server	11
13	Настройка, отображающая колонку таблицы («Сервера»), генерируемую из динамического наименования сервера.	[Column:UI\Properties\Servers\Table\Name] Source = * Type = String Default = Server	12
	Настройка, отображающая статическую колонку таблицы.	[Column:UI\Properties\Servers\Table\Server] Source = Server Type = String	-

Таким образом, мы полностью определили необходимые от пользователя

настройки по динамическому формированию диалогов изменения файла настроек сервера «FIX».

Определим способы формирования и доступа к возможностям редактирования и создания новых записей в таблицах отображающих настройки серверов «FIX».

Для доступа к этим возможностям будем использовать контекстное меню. Такое меню, должно вызываться по нажатию правой клавишей (при нажатии на существующую запись – открывается контекстное меню с возможностью редактирования, при нажатии на пустое поле открывается контекстное меню с возможностью создания новой записи).

В данное меню удобным будет добавить возможность «выделить все», позволяющую быстро выделить все существующие элементы для последующего экспорта.

Для формирования диалога создания \ редактирования записей таблиц «Сервера», «Сессии» будем пользоваться набором правилом:

1) Определять интерфейсы ввода данных в том порядке, в котором они объявлены в файле настроек «MFBM.ini» (также определяется порядок колонок в таблице);
2) Интерфейсы должны ограничивать действия пользователя по вводу (если используется тип данных перечисление, то организовывать выпадающий список, если тип данных строка - то текстовое поле).

Определим графическое представление диалога «подключения» к серверам «FIX». Для этого составим его схематическое представление.

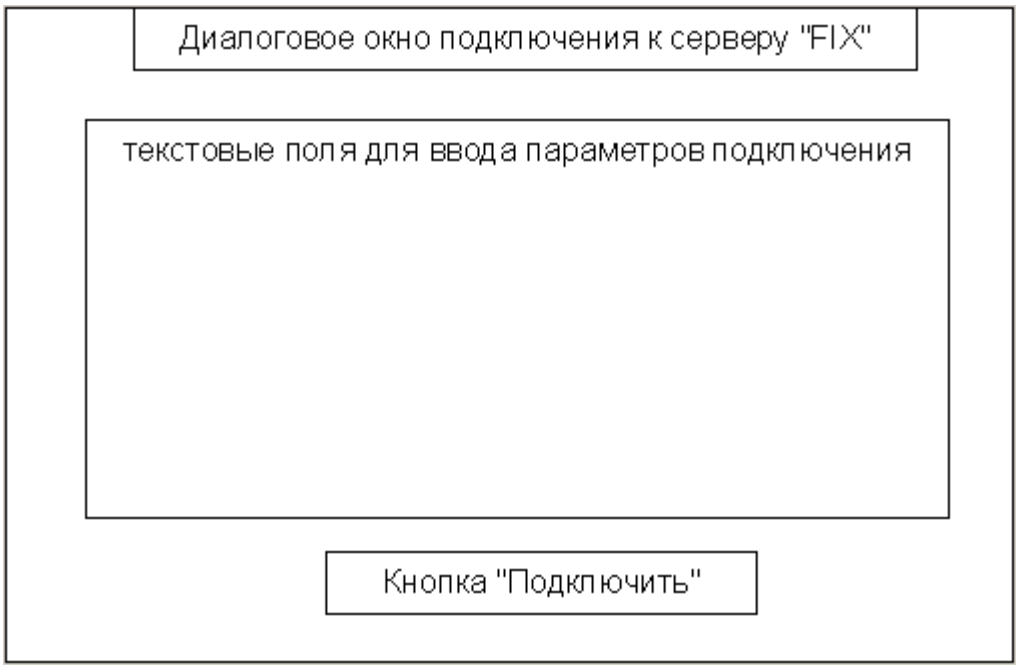


Рисунок 2.3. Схема диалогового окна подключения к «FIX» серверам.

Диалоговые окна такого типа широко используются во многих приложения (клиентские приложения к базам данных и др.).

Определим схему диалогового окна для отключения от подключенных серверов «FIX».

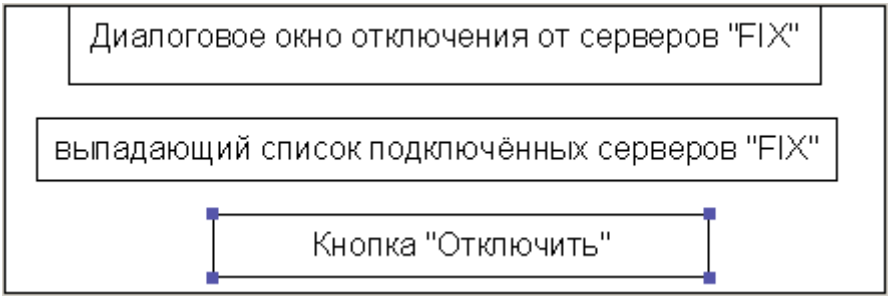


Рисунок 2.4. Схема диалогового окна отключения от «FIX» серверов.

Таким образом, мы полностью поддержали спроектированные ранее решения по способам доступа ко всем возможностям проектируемого приложения.

Для отправки сообщений пользователям “FIX” сервера, мы должны

отправлять следующие данные: идентификационные параметры пользователя (пользователей), тему сообщения, текст сообщения.

Для этого составим диалог, который содержит:

1. поле, отображающее список пользователей кому будет отправлено сообщение через символ сточки с запятой;
2. поле для ввода строковой переменной;
3. поле для ввода текстовой переменной;
4. кнопку отправки сообщения.

Спроектировав данный диалог, мы полностью закончили проектирование пользовательского интерфейса для проектируемого приложения.

2.4. Проектирование модуля работы с «XML» сообщениями.

Спецификация «FIX» протокола определяет способ передачи сообщений «FIXML», для передачи контекстной информации в “XML”. Именно такой способ, используется для отдачи команд серверу. Для этого используется сообщение идентифицируемое тегом «35=n». Передаваемая команда отправляется в тегах «212» и «213» («XmlDataLen» и «XmlData» соответственно). Таким образом, нашей задачей становится идентификация группы вызываемых функций и возможных ответов от них, для проектирования модуля инкапсулирующего работу с “XML”, что позволит остальным модулям пользоваться готовыми строковыми переменными.

Задачами такого модуля становятся:

1. предоставить набор функций, генерирующих определённые “XML” сообщения, обрабатываемые другими модулями;
2. инкапсулировать все действия касающиеся работы с «XML»;

Для работы с «XML» мы воспользуемся библиотекой “Xerces-C” данная

библиотека позволяет, как конструировать “XML” сообщения, так и анализировать полученные с проверкой корректности составления.

Библиотека требует предварительного вызова метода, который конструирует статические переменные библиотеки (и вызова метода которые уничтожает эти переменные после всех необходимых действий).

Для этого мы разместим данные операции в конструкторе и деструкторе класса инкапсулирующего действия с преобразованием «XML» сообщений.

Создадим функцию конструкции необходимых элементов библиотеки («DOM» генератор, и «DOM» анализатор). Общая структура разрабатываемых функций:

1. название функции содержит проделываемое действие (упаковывание в «XML», или распаковывание из «XML») и название функции вызываемой на сервере;
2. каждая функция упаковывания должна принимать необходимые параметры для генерации корректного сообщения;
3. каждая функция распаковывания должна возвращать массив пар, «наименование параметра ответа» - «значение», такой массив позволит полностью идентифицировать ответ в других модулях библиотеки;

Определим общую структуру работы модуля.

Пользователь модуля должен вызывать функции для получения «XML» сообщения готового к отправке, или для расшифровки результата, полученного в результате отправки. Такой результат может содержать информацию о том, что запрос был составлен не верно. Для этого пользователи функции распаковки должны будут проверять возвращаемый результат, в котором в соответствии строке «ResultCode» должен устанавливаться:

1. «0» при успешной обработке;

2. «номер ошибки» при неуспешной обработке, и добавочное описание ошибки в соответствие к «Description».

Предоставим схему класса, реализующего работу данного модуля.

Рисунок 2.5. Схема класса реализующего модуль «Работа с ‘XML’ сообщениями».

2.5. Проектирование модуля общения с программными серверами «FIX».

Для корректной реализации соединения и поддержки сессий протокола «FIX», мы будем пользоваться библиотекой «B2BITS Fix-Antenna». Данная библиотека позволяет использовать готовые решения для администрирования «FIX» серверов. Поэтому в нашу задачу входит определения следующего набора функций:

- Создание подключения к серверу «FIX»;
- Отправка сообщения к серверу «FIX»;
- Получение ответа от сервера «FIX»;
- Отключение от сервера «FIX»;

Условимся, что после отправки сообщения модуль обязан ожидать ответа, и только после получения ответа вернуть результат. Таким образом, приложение, использующее данный модуль, сможет определять вернула ли функция ответ, и на какой запрос этот ответ.

Данный модуль должен возвращать принимать от клиента уже готовое «XML» сообщение, упаковывать его в соответствующее «FIX» сообщение. А при получении «FIX» сообщение автоматически распаковывать «XML» сообщение.

Для создания удобного доступа пользователю требуется создать объединяющий модуль, который инкапсулирует работу с модулями работы «XML» и данным модулем работы с «FIX», тогда пользователь сможет использовать один класс, как для подключения, отключения так и для отправки сразу готовых функций команд. Которые принимают в качестве параметров необходимые значения переменных и возвращают массив пар строк «название возвращаемого параметра», «значение параметра».

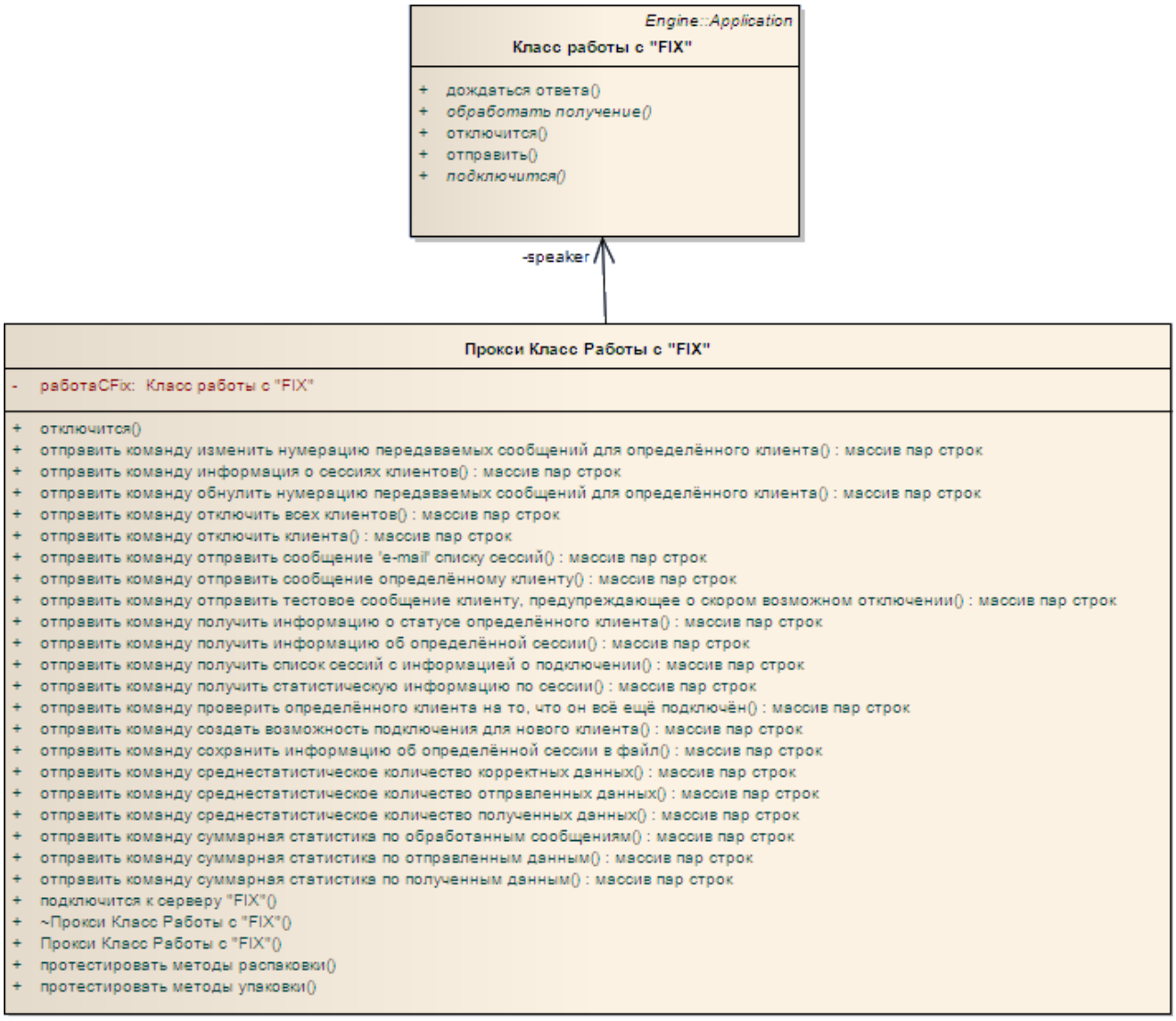


Рисунок 2.6. Диаграмма модуля работы с «FIX»

Таким образом, данный модуль представляет собой законченный и полноценный модуль, инкапсулирующий всё необходимое для управления серверами «FIX». Такой модуль полностью абстрагирует его пользователя от управления созданием, пересозданием, удалением, очисткой памяти при работе с ‘XML’, ‘FIX’. Пользователю такого модуля достаточно использовать следующий алгоритм работы с модулем:

1. Создать экземпляр класса «Прокси Класс Работы с ‘FIX’»;
2. Вызвать функцию подключения, с параметрами подключения;
3. Вызвать функцию «отправить команду на сервер …»;

4. Обработать ответ;
5. …
6. Вызвать функцию «отправить команду на сервер …»;
7. Обработать ответ;
8. Вызвать функцию отключения от сервера «FIX»;
9. Удалить экземпляр класса «Прокси Класс Работы с ‘FIX’»;

Создаваемый класс, должен сам определять количество используемых сессий, и создавать или удалять элементы библиотек «B2BITS Fix-Antenna», «Xerces-C».

Для этого данный класс должен использовать возможности приёмов подсчёта ссылок, и скрывания настоящих конструкторов и деструкторов, давая доступ к инициализации реальных классов подключаемых к серверам «FIX» только через свои функции.

Это означает, что класс «Прокси Класс Работы с ‘FIX’» должен:

- при вызове функции конструктора увеличивать значение статической переменной (переменной класса), и при пересечении границы «0-1» конструировать элементы необходимые для работы используемых библиотек;
- при вызове функции деструктора уменьшать значения статической переменной, и при пересечении границы «1-0», уничтожать все ранее созданные элементы необходимые для работы используемых библиотек.

Задачей пользователя такого модуля, выделить и настроить работу потоков системы, для управления таким модулем. Включая корректные обработки возникающих исключений в результате потери соединения, или в результате нехватки памяти компьютера.

2.6. Проектирование взаимодействия между модулями.

После проектирования всех необходимых модулей, которые реализуют необходимое количество функций предъявляемых к администраторскому приложению управления серверами «FIX» можно приступить к заключительному этапу проектирования. Проектированию взаимодействия, определяющему весь необходимый объем работ, которые должны быть проделаны программистом для реализации проектируемой утилиты.

Для определения как должны взаимодействовать модули, нужно чётко определить ходи работы приложения и порядок вызываемых действий.

Для этого требуется создать диаграммы последовательностей действий пользователя определяющих взаимодействие пользователя и созданных подсистем. Представим её на рисунке 2.7. Данная диаграмма показывает порядок работы модулей в системе, а значит, определяет их взаимосвязи.

Выделим правила запуска модулей и взаимодействия между ними:

1) Первым запускается модуль пользовательского интерфейса. Данный модуль загружает все необходимые библиотеки для отображения графического представления (главного окна) администраторской утилиты «MFBM»;
2) Модуль пользовательского интерфейса создаёт класс запускающий модуль работы с «FIX», данный модуль автоматически запускает модуль работы с ‘XML’; данные действия производятся только в случае, когда пользователь приложения пытается подсоединиться к серверу (серверам) «FIX»;
3) Модуль пользовательского интерфейса удаляет класс останавливающий модуль работы с «FIX» (автоматическая остановка модуля работы с ‘XML’); данные действия производятся только в

случае, когда пользователь приложения отключается от всех “FIX” серверов.

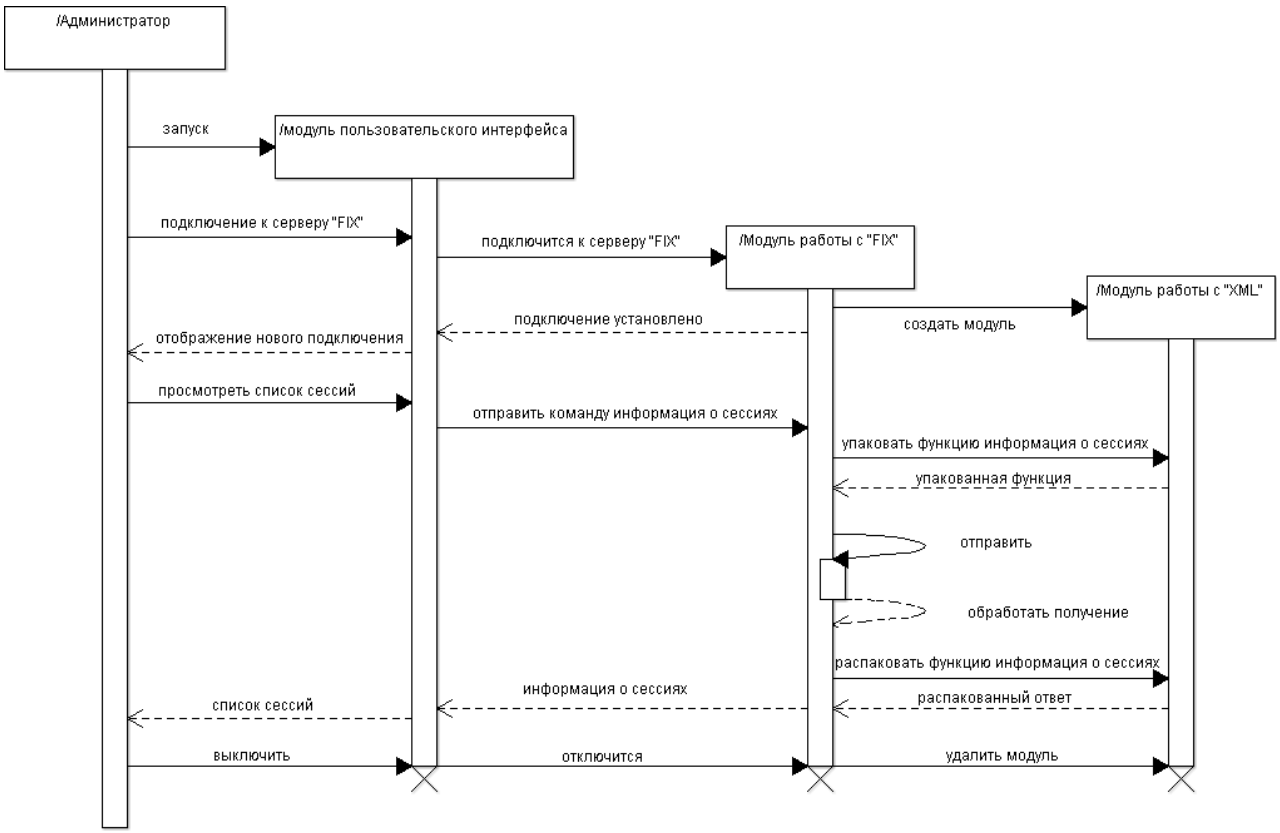


Рисунок 2.7. Диаграмма последовательности действий администратора сервера «FIX».

Структуру взаимодействия модулей, изображенную на рисунке 2.8 можно описать следующим образом:

1) Модуль пользовательского интерфейса, содержит все необходимые интерфейсы по управлению приложением, и начинает работу, заканчивает работу, оперирует с возможностями других модулей системы.

2) Модуль работы с «FIX» полностью инкапсулирует как свою работу, так и работу с модулем работы с «XML». Это позволяет добиться простой структуры приложения использующего такой модуль. Такой модуль

лучше выносить в отдельную библиотеку.

3) Модуль работы с «XML» является полностью независимым, его задачей становится инкапсуляция своей работы. Для нашей задачи может быть включен в отдельную библиотеку предоставляющую доступ к функциям серверов «FIX».

Рисунок 2.8. Диаграмма компонент
администраторской утилиты “MFBM”

3. РЕАЛИЗАЦИЯ АДМИНИСТРАТИВНОЙ УТИЛИТЫ УПРАВЛЕНИЯ ПРОГРАММНЫМИ СЕРВЕРАМИ «FIX»

3.1. Разработка объектной модели реализующей административную утилиту.

Создание приложения подразумевает под собой прохождение некоторых этапов. Последним этапом в таком процессе становится «разработка». В данном разделе нашей целью становится описание причин разработки полученной объектной модели.

Итак приложение использует стандартный тип модели подчинения, когда требуется отдельная обработка графического содержимого с которым работает пользователь. Данная подсистема решает следующие вопросы:

1. управление остальными подсистемами (включая подсистему очереди запросов);
2. быстрое реагирование на действия пользователя.

Подсистема очереди запросов отвечает за корректную работоспособность следующих задач:

1. корректное управление другими подсистемами по командам подсистемы графического содержимого;
2. создание очереди запросов пользователя, работа с очередью запросов пользователя.

Данные подсистемы реализованы в двух классах «CAsyncXMFBM» и «CMFBMDlg». Класс «CasyncXMFBM» отвечает за корректную работу подсистемы очереди запросов. А класс «CMFBMDlg» отвечает за корректную обработку действий пользователя. Как видно на рисунке 3.1 класс представляющий возможности графической работы с пользователем имеет множество методов. Часть этих методов ('OnFTPWrite', 'OnMicexBridgeConnect'

и т.д.) представляют собой методы вызываемые впоследствии действий пользователя. А часть методов ('deletePage', 'deleteFixSession' и т.д.) действуют в результате возврата результатов от использующих их подсистем.

Класс «CAsyncXMFBM» позволяет асинхронно отправлять команды на сервера, а при получении ответа возвращать его в приложение.

Рисунок 3.1. Диаграмма классов реализующих много-поточность приложения "MFBM"

1. OnClose() - функция окончания работы приложения;
2. OnFileLoad() - функция вызываемая при попытке загрузки файла настроек сервера «FIX» по протоколу «FTP»;
3. OnFileSave() - функция вызываемая при попытке сохранения файла настроек сервера «FIX» по протоколу «FTP»;
4. OnFixSessionDelete() - функция вызываемая при желании пользователя отключить определённого клиента от сервера «FIX»;
5. OnMicexBridgeConnect() - функция вызываемая при желании пользователя подключится к новому серверу «FIX»;
6. OnMicexBridgeConnectAll() - функция вызываемая при желании пользователя подключится ко всем серверам «FIX» заданным в файле настроек;
7. OnMicexDisconnect() - функция вызываемая при желании пользователя отключится от конкретного сервера «FIX»;
8. sendEmails() - функция вызываемая когда пользователь пытается отправить сообщения клиентам «FIX» сервера.

При работе с приложением могут возникать множества различных исключительных ситуаций, таких как «невозможность подключится к 'FTP' серверу», «невозможность подключится к 'FIX' серверу», данные исключительные ситуации отображают перед пользователем либо с помощью специальных информационных диалоговых окон, либо с помощью специальных статусов сообщений. Отображаемых в поле статуса главного окна.

Класс «CasyncMFBM» это класс предоставляющий возможности асинхронной работы. Основные функции по управлению отдельным потоком и управлением очереди задач находятся в классе «CAsyncX», данный класс

предоставляет набор функций для складывания этих задач в очередь:

1. fixConnect();
2. fixConnectAll();
3. FixDeleteSession();
4. fixDisconnect();
5. fixSendEmails();
6. fixGetSessions();
7. readFTPFile();
8. writeFtpFile();

Данные методы позволяют упаковывать в очередь команды отдаваемые пользователем при работе с администраторской утилитой «MFBM».

Аналогичные используемые методы в названии которых используется подчёркивание, это методы которые работают в асинхронном потоке. Такие методы вызываются в асинхронном потоке и возвращают ответ в вызывающий поток посредствам системы сообщений.

3.2. Разработка модуля работы с «XML» сообщениями.

Модуль работы с «XML» сообщениями должен уметь упаковывать команды и распаковывать ответы для дальнейшей из обработки. Для этого был разработан класс «CommandXmlTranslator» предоставляющих набор методов для запаковывания команд «packНазваниеКоманды» полностью соответствующий командам определённым на этапе разработки, и набор функций для распаковывания ответов на команды от сервера «unpackНазваниеКоманды».

Рисунок 3.2. Диаграмма класса реализующего модуль работы с "XML".

Класс также содержит набор строковых идентификаторов определяемых

при запуске приложения константными строковыми константами. Эти строковые константы отсылаются на «FIX» сервер вместе с каждой командой, чтобы в последствии идентифицировать возвращаемое значение.

Данный класс обязан быть создан с помощью функции инициализации Init(), которая создана для ограничения количества экземпляров класса. Данная защита сделана, чтобы пользователь не мог создать в памяти больше одной копии экземпляра, что позволяет защитить используемые внутри класса строковые буфера памяти для расшифрования сообщения от вмешательства другими экземплярами, а также экономит память используемую пользователем.

Предполагается, что пользователь будет следить за тем, чтобы элементы возвращаемого объекта были защищены от вмешательства разными потоками одновременно.

Уничтожение объекта класса происходит при вызове функции «Release()».

Правильным ходом работы с этим классом является следующая последовательность действий:

1. вызов метода инициализации «Init()» и сохранения в переменную адреса созданного объекта;
2. вызов метода запаковки сообщения (например «packDelete()»);
3. отправка сообщения с помощью модуля работы с программными серверами «FIX»;
4. получение ответа с помощью модуля работы с программными серверами «FIX»;
5. вызов метода распаковки ответа (например «unpackDelete()»);
6. вызов метода «Release()» для очистки памяти от созданного объекта.

Объект может быть использован долгое время для работы с «XML» сообщениями.

3.3. Разработка модуля работы с программными серверами «FIX».

Задачами данного модуля являются:

1. корректная работа с «FIX» серверами;
2. инкапсуляция работы «XML» модуля, для предоставлению пользователям этого модуля, открытого набора функций доступных из одного места (например методы одного объекта).

Опишем соответствия классов «CommandXmlTranslator», «CAdminMicex», «Speaker» изображенных на рисунке 3.2:

1. «Прокси Класс Работы с ‘FIX’» соответствует классу «CAdminMicex», данный класс скрывает действия модуля работы с «FIX» и модуля работы с «XML»; в результате пользователю необходимо обращаться только к одному классу, который позволяет использовать оба модуля через один интерфейс.

2. Модуль для работы с «XML», соответствует классу «CommandXmlTranslation»;

3. класс для работы с «FIX», соответствует классу «Speaker».

Опишем работу класса «Speaker». Класс «Speaker» отвечает за корректное взаимодействие разработанной утилиты с сервером «FIX». Для этого класс реализует все необходимые методы для работы библиотеки «B2Bits Fix-Antenna».

Данный класс предоставляет доступ к следующим возможностям: подключение к серверу, отключение от сервера, отправка сообщения серверу, принятие ответа от сервера.

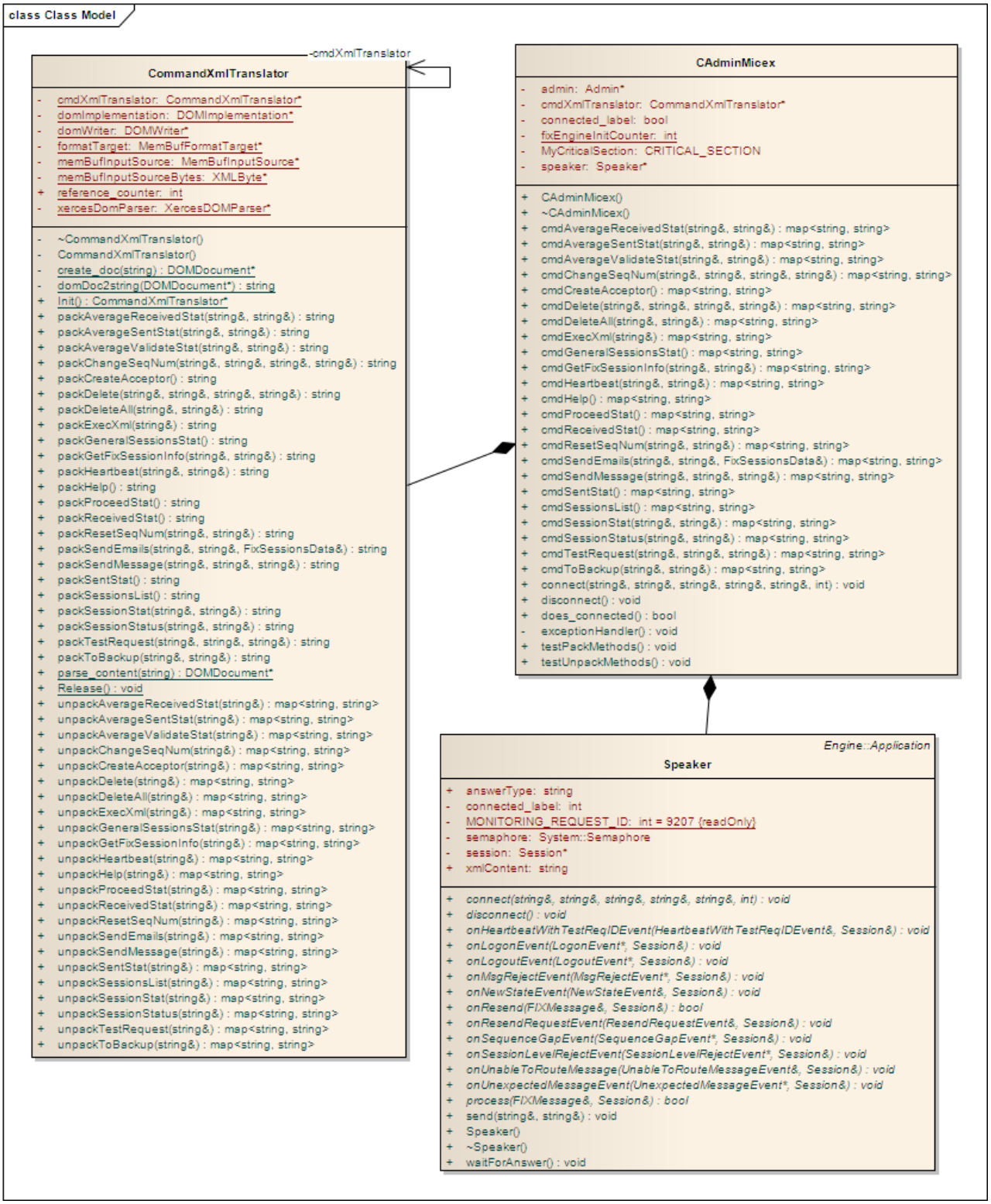


Рисунок 3.3. Диаграмма взаимодействия классов модуля работы с "FIX"

Класс для предоставления доступа использует методы:

1. connect() - для подсоединения к новому «FIX» серверу;
2. disconnect() - для отсоединения от сервера;
3. send() - для отправки сообщения на сервер «FIX»;
4. process() - для получения ответа с сервера.

Для корректной работы пользователь должен вызывать метод отправки сообщения только после подключения и перед отключением, в противном случае будет использована исключительная ситуация (выброс исключения языка C++).

Таким образом, класс «CAdminMicex» позволяет пользователю сразу использовать возможности модуля работы с «FIX» и модуля работы с «XML», для этого класс «CAdminMicex» предоставляет набор методов являющихся совокупностью методов предоставляемых обоими классами.

Такие методы можно поделить на следующие виды:

- методы для соединения и отсоединения;
- методы для отправки команд на сервер «FIX»;

Методы для соединения и отсоединения, полностью пересылают набор параметров в соответствующие методы класса «Speaker»;

Методы для отправки команд на сервер «FIX» используют следующий алгоритм работы:

1. упаковка переданных параметров в текстовый формат «XML» сообщения;
2. генерация «FIX» сообщения на основе «XML» сообщения;
3. отправка «FIX» сообщения на сервер «FIX»;
4. ожидание ответа на отправленное сообщение;

5. получение ответа от «FIX» сервера;
6. распаковка «FIX» сообщения, получения текстового представления «XML»;
7. проверка и распаковка «XML» сообщения;
8. возвращение результата пользователю в виде массива пар строк.

Данный алгоритм позволяет сделать операцию отправки сообщения и получения ответа незаметной для использующих такой модуль. Со стороны использующих данный модуль всё выглядит как будто вызывается обычная функция и возвращается результат.

Данный набор классов был выделен в отдельную динамическую библиотеку, что позволяет очень просто, как тестировать корректность работы готовых модулей отдельным приложением, так и использовать в качестве отдельного модуля приложения.

3.4. Разработка модуля пользовательского интерфейса.

Модуль пользовательского интерфейса подсистемы работы с пользователем и подсистемы асинхронного выполнения команд пользователя. Это взаимодействие представлено на рисунке 3.3.

Опишем классы представленные на данном рисунке:

1. класс «CMFBMApp» является классом входа в приложение, он содержит главную функцию запускающую и отображающую главное окно;
2. класс «CWarningDialog» отвечает за отображение информационного окна предупреждения пользователя;
3. класс «CDisconnectDialog» отвечает за отображение диалога отключения административной утилиты от сервера (аналогичное действие у диалога подключения к серверу);

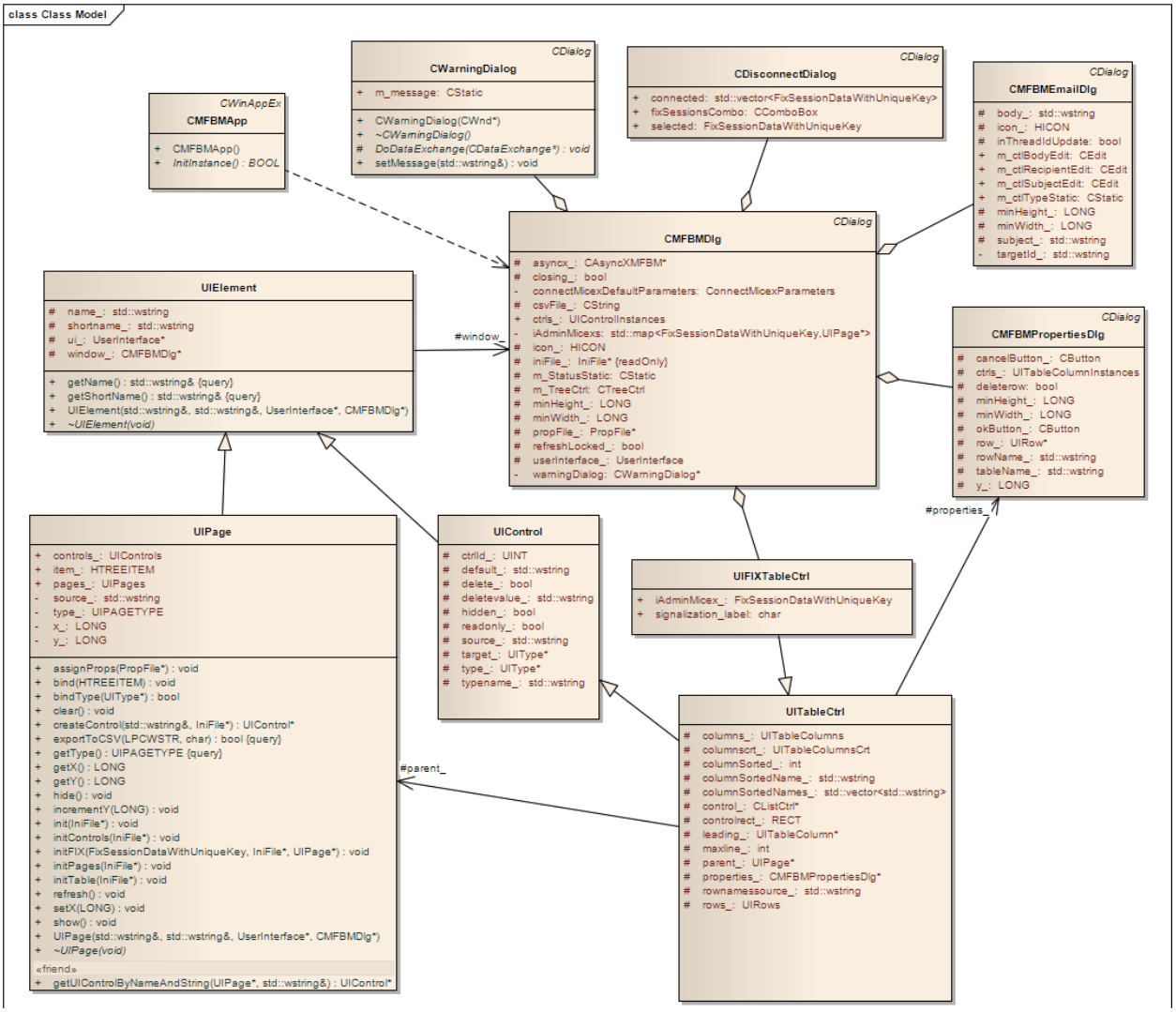


Рисунок 3.4. Диаграмма классов реализующих модуль пользовательского интерфейса.

4. класс «CMFBMEmailDlg» отвечает за отображения окна позволяющего пользователю вводить «e-mail» сообщение и тему такого сообщения для отправки выдранным пользователям;
5. класс «CMFBMDlg» отвечает за отображение главного окна приложения;
6. класс «CMFBMPropertiesDlg» зарезервирован для дальнейшего развития системы, если потребуется изменять настройки на лету, а также используется для динамических диалогов изменения параметров полученных по «FTP»;
7. класс «UIElement» представляет собой базовый класс динамических

элементов отображающихся на главном окне;

8. класс «UIPage» представляет собой базовый класс для динамических страниц, которые пользователь может добавлять в приложения посредствам изменения файла настроек «MFBM.ini»;
9. класс «UIControl» специфицирует отдельный динамический элемент управления для ввода или отображения той или иной информации;
10. класс «UITableControl» представляет собой отдельный динамический элемент управления для ввода или отображения информации в табличном виде;
11. класс «UIFixTableCtrl» представляет собой отдельный элемент управления отображающий список сессий получаемый с подключенных серверов.

На данный момент класс «CMFBMDlg» является основным классом системы работающим с пользователем, он содержит доступ:

- диалоговым окнам подключения, отключения «FIX» серверов;
- к диалоговому окну изменения настроек текущей таблицы;
- к информационному диалоговому окну отображающему информацию о изменении состояния подключения;
- к табличным данным выраженным в виде страниц хранимых в виде динамического массива элементов «UIPage».

Все эти классы позволяют специфицировать динамичность частей интерфейса которые должны быть динамичными и статичность интерфейса который должен быть статическим.

Такое поведение, позволяет добавлять новые элементы динамичного интерфейса простым наследование и изменением специфики существующих

элементов динамически отображающиеся во время работы административной утилиты.

3.5. Тестирование работы административной утилиты управления программными серверами «FIX».

Тестирования администраторской утилиты подразделяются на два варианта:

1. само-тестирование специально предназначенными для этого функциями ('testPackMethods()', 'testUnpackMethods()' класса «CAdminMicex» представленного на рисунке 3.3);
2. тестирование при помощи создания различных условий запуска, различных настроек подключения, различных настроек использования графических данных.

Рассмотрим первый вариант тестирования. Само-тестирование (тестирование во время написания) хорошо известный метод разработки программного обеспечения. Данный метод позволяется убедится, что в данный момент функции (участки программ) работают корректно. При возникновении каких-либо ошибок. Новые входные данные добавляются к существующим тестовым для дальнейшего использования.

Данный метод тестирования мы использовали при разработке модуля работы с «XML». Для этого мы создали два дополнительных тестовых метода, которые могут вызываться любым приложением. Первый метод отвечает за корректную работу функций упаковки параметров в «XML» сообщение, второй метод отвечает за корректную работу функций распаковки параметров в «XML».

Для тестирования корректности работы с множественными серверами система была настроена на 4-ре различных сервера одновременно.

Тестирование показало, что разработанная утилита может успешно работать с несколькими серверами одновременно. При этом пользователь может использовать все её возможности касающиеся работы с каждым из подключенных серверов.

Для тестирования корректности работы с «FTP» серверами. Приложение было запущено и работало с двумя различными «FTP» серверами. Ни на одном из них проблем с работой не возникло (тестировались также требования ввода имени и пароля пользователя для доступа к «FTP»).

Таким образом, можно сказать что приложение успешно прошло тестирование, и подготовлено для работы с конечным пользователем

ЗАКЛЮЧЕНИЕ

В результате работы мы получили готовое для использования приложение администрирующее сервера «FIX». Приложение позволяет качественно и быстро выполнять свои задачи использующему его администратору серверов.

Главным отличием разработанного приложения «MFBM», является его пользовательский интерфейс и способ подачи информации. Графическое представление табличных данных «XML» сообщения, позволяет пользователю получать всю необходимую ему информацию.

Возможность настройки приложения под конкретные сервера «FIX» позволяет не только настраивать доступ к серверам, но и графическое представление получаемых данных. Это позволяет адаптировать приложение к самым различным серверам. Приложение обладает возможностью работы с несколькими серверами одновременно и возможностью оповещения пользователя о потере соединения с определённым сервером. Что позволяет администратору серверов своевременно реагировать на различные неполадки, возникающие с оборудованием, или программным обеспечением.

Все эти свойства делают разработанное приложение конкурентно-способным, удобным в использовании и позволяет администратору получать новые способы для управления серверами «FIX».

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1) Официальный сайт «FIX» протокола. Раздел документации. [Электронный ресурс] / “FIX Protocol Limited”, 2009. – Режим доступа: http://www.fixprotocol.org/specifications/FIX.4.4, свободный. – Яз. Англ.

2) Официальный сайт библиотеки «B2BITS Fix-Antenna». Раздел документации. [Электронный ресурс] / “EPAM Systems”, 2009. – Режим доступа: http://corp-web.b2bits.com/fixacpp/doc/html/, свободный. – Яз. Англ.

3) Официальный сайт библиотеки «Xerces-C». Раздел документации. [Электронный ресурс] / “The Apache Software Foundation”, 2009. – Режим доступа: http://xerces.apache.org/xerces-c/api-3.html, свободный. – Яз. Англ.

4) Страуструп Б. Язык программирования C++. Специальное издание. / Б. Страуструп - Т.: Бином, 2008. - 1104с.

5) Керниган Б., Ритчи Д. Язык программирования С. / Б. Керниган, Д. Ритчи - М.: Вильямс, 2008. - 304с.

ПРИЛОЖЕНИЕ А

Руководство оператора административной утилиты управления программными серверами «FIX».

1. Описание приложения.

Главное окно приложения состоит из двух областей (рис. 1):

- дерева страниц (слева);
- представления выбранной страницы (справа).

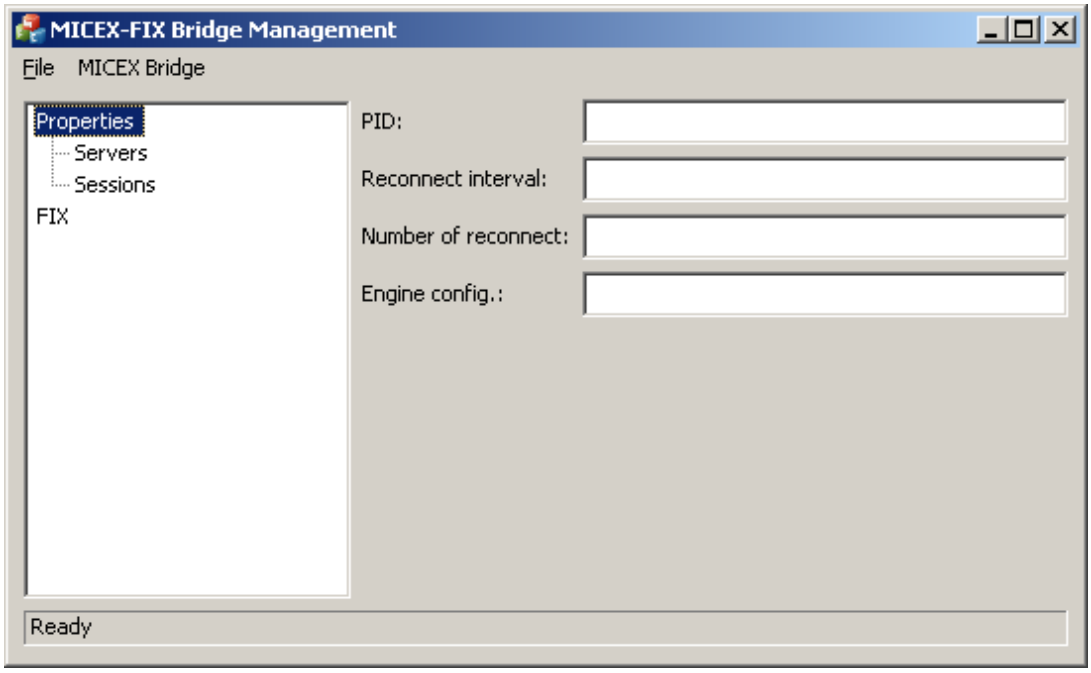


Рис. 1. Интерфейс приложения MFBM

Дерево страниц и элементы управления в их представлении формируются как во время загрузки приложения, так и во время подключения к серверам приложения. Данная информация читается из файла «MFBM.ini», который должен располагаться в том же каталоге, что и исполняемый файл программы («MFBM.exe»).

Содержимое файла «MFBM.ini» представляет собой набор секций и ключей, принадлежащих секциям. Секция формируется строкой “[<имя секции>]”. Ключ - <имя ключа>=<значение>. Комментарии начинаются символом # и заканчиваются концом строки.

Настройки интерфейса хранятся в секциях: UI, Types, List:<имя списка>, Enum:<имя перечисления>, Page:<полное имя страницы>, Control:<полное имя

элемента>, Column:<полное имя колонки>.

Каждая из страниц может содержать либо таблицу, либо набор элементов диалога.

1.1. Общие возможности таблиц.

Таблицы любого типа позволяют копировать выбранные строки в буфер обмена с помощью команды “Copy”.

Команда “Export” позволяет сохранять данные любой таблицы в “CSV” файл. Сохраняются все данные таблицы вне зависимости от того, выбраны какие-то строки или нет. В качестве разделителя используется первый символ ключа CSV секции UI (по умолчанию запятая «,»). Если в какой-либо ячейке таблицы встречается символ-разделитель, он заменяется на пробел. Следует учесть, что MS Excel использует различные разделители в CSV файлах в зависимости от региональных настроек системы (Control Panel\Regional and Language Options). Так если параметр стандарты и формат установлен Russian (где разделитель дробной и целой частей числа запятая), то в качестве разделителя Excel использует точку с запятой «;».

1.2. Таблица основного файла настроек сервера.

Таблицы позволяют выбирать одну, несколько или все записи. Для действий с выбранными записями необходимо использовать контекстное меню (вызывается нажатием правой кнопки мыши). Таблицы, относящиеся к файлу .properties, позволяют удалять, редактировать и добавлять записи.

1.3. Таблицы подключённых серверов.

Данные таблицы, появляются в результате подключения к серверу (см. п. 1.5).

Таблица позволяет:

- отправлять сообщение группы «C» (‘E-Mail’ сообщение) выбранным сессиям;
- отключать выбранные сессии.

1.4. Описание настроек файла «MFBM.ini».

1.4.1. Секция [FTP].

Данная секция отвечает за настройку FTP сессии к серверу, хранящему файл настроек («MCX-Fix.properties»).

Параметрами секции являются:

- «Address» – имя или IP адрес FTP сервера;
- «User» – имя пользователя FTP сервера (пользователь должен иметь права на чтение и запись файлов);
- «Password» – пароль указанного пользователя (если этот ключ не будет указан, то при подключении к FTP серверу, программа предложит ввести логин и пароль);
- «File» – полное имя файла для редактирования;

EOL – терминатор строки для редактируемого файла (используется только при сохранении файла). Возможные значения LF, CR, LFCR и CRLF. (По умолчанию используется CRLF).

1.4.2. Секция [MFDB_DEFAULT].

Данная секция отвечает за настройку стандартных параметров выводимых в диалоговом окне «Connect To Micex Data», а также за настройку параметров подключения по умолчанию к серверам Micex Bridge.

Данные параметры выводятся в диалоговом окне, и возможны для изменения пользователем в ходе работы приложения.

Подробнее диалоговое окно «Connect To Micex Data» описано в пункте 1.5.

Параметры секции:

- SenderCompId – идентификатор клиента;
- TargetCompId – идентификатор административной сессии сервера;
- Username – имя пользователя административной сессии;
- Password – пароль административной сессии;
- Host – адрес сервера;
- Port – порт сервера;
- Timeout – интервал обновления списка клиентов в миллисекундах (при активном контекстном меню для таблицы FIX или при выполнении загрузки или сохранения файла по ftp очередное обновление списка сессий будет пропущено).

Также данные параметры используются в качестве параметров по умолчанию для настроек остальных секций, описывающих параметры серверов. (см. п. 1.5).

Остальные секции настраивают графический интерфейс приложения MFBM. Просьба не изменять данные параметры без консультации команды разработчиков.

1.5. Параметры настройки сессий подключаемых по нажатию «Connect All…».

1)Для настройки сессий подключаемых при нажатии “Connect All…” используется секция файла настроек «MFBM.ini» под названием **“[MFDBS]**”.

2)Данная секция должна содержать строку настроек:

3)Servers = <список уникальных идентификаторов серверов MICEX Bridge разделённых символом «запятая»>, например:

4)[MFDBS]

5)Servers = Production, 194_clientStand, 194_clientStand2

6)Данные строки настраивают приложение на автоматическое подключение к трём серверам.

7)Для уточнения параметров подключения используются секции, имена которых совпадают с уникальными идентификаторами. Пример такой секции:

8)[Production]

9)SenderCompId = MonitoringTool1

10) TargetCompId = FIXADMIN

11) Username = MonitoringTool

12) Password = MonitoringToolPassword

13) Host = 10.6.3.35

14) Port = 9111

15) Каждый параметр не является обязательным для ввода (используются параметры по умолчанию из секции **«[MFDB_DEFAULT]**»).

1.6. Диалоговое окно «Connect To Micex Data».

Данное диалоговое окно (см. рис. 2) используется для подключения к серверам MICEX Bridge.

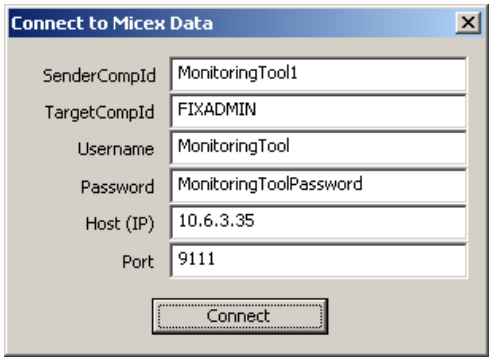


Рис. 2. Диалоговое окно «Connect To Micex Data».

По умолчанию (при первом запуске диалогового окна) параметры заполняются из файла настроек (MFBM.ini см. п. 3.4.2). Данные параметры допускается изменять для соединения с другими серверами.

При заполнении параметров требуется учитывать настройки серверов находящиеся в файлах “engine.properties” (секция “Monitoring”). При настройке серверов требуется учитывать. Что FIX протокол не позволяет подсоединение к различным серверам с одинаковой парой значений «SenderCompID-TargetCompID». Поэтому такие пары значений должны быть различны для каждой администраторской сессии, каждого сервера MICEX Bridge.

1.7. Диалоговое окно «Disconnect:».

Данное диалоговое окно предназначено для разрыва соединения с серверами MICEX Bridge. Для отключения от сервера необходимо:

- выбрать в выпадающем списке значение идентифицирующие сервер;
- нажать кнопку «Ok».

См. рис. 3.

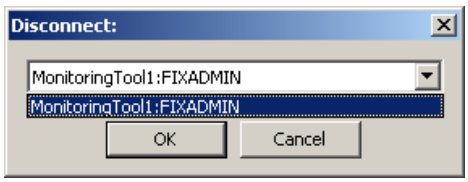


Рис. 3. Диалоговое окно «Disconnect:»

1.8. Диалоговое окно «E-mail».

Данное диалоговое окно предназначается для отправки сообщений «35=C» (E-Mail) сообщений одному или группе пользователей MICEX Bridge сервера. (см. рис. 4).

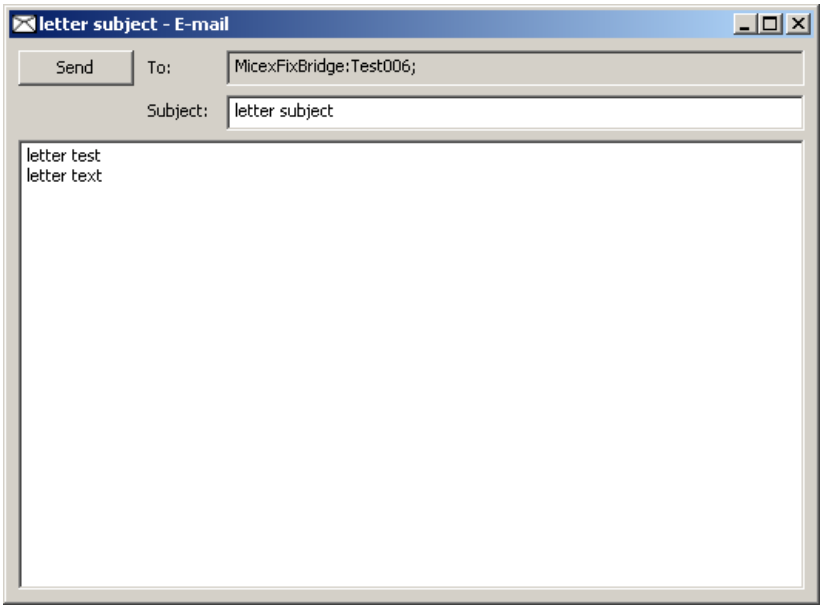


Рис. 4. Диалоговое окно отправки “E-mail”.

Для появления данного диалогового окна требуется из таблицы подключенного сервера выбрать FIX сессии, вызвать контекстное меню, выбрать пункт “Send Email…”

2. Часто задаваемые вопросы:

2.1. Почему не подключается к серверу MICEX Bridge?

Возможными проблемами могут стать:

- неверный набор параметров подсоединения (проверьте правильность настроек соединения в диалоговом окне “Connect To Micex Data”);
- отсутствие доступа к серверу по данному IP, Port (проверьте, запущен ли сервер и доступен ли он по данному IP, Port);
- высокая нагрузка приложения MFBM при обновлении сессий уже подключенных серверов, или скачиванию данных по FTP (подождите некоторое время и попробуйте запустить операцию снова);
- набор параметров подключения (а именно SenderCompID, TargetCompID) уже используется при соединении с другим сервером (неверно настроены сервера MICEX Bridge).

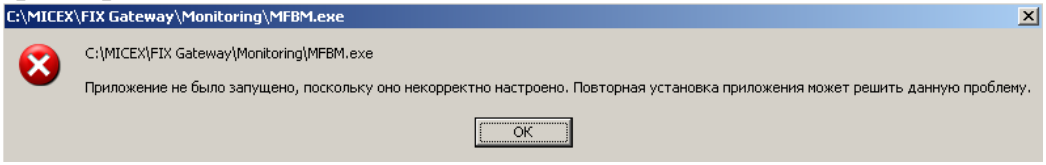
2.2. Что писать в файл настроек MFBM.ini для корректной загрузки/сохранении файла настроек на FTP сервере?

Требуется проверить наличие возможности соединения с данным сервером любым ftp-клиентом. Если возможность соединения существует: то требуется ввести полный путь к файлу настроек (включая имя файла настроек) MCX-Fix.properties.

2.3. Почему при нажатии ConnectAll, один из серверов стал подключаться к другому серверу, или перестал подключаться к серверу, хотя изначально приложение подключалось к серверу без каких-либо проблем?

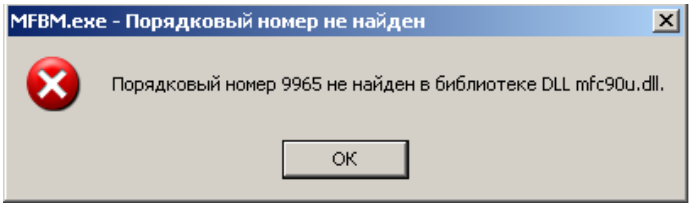
Если данная проблема произошла после использования «Connect…», то требуется внимательно посмотреть файл настроек MFBM.ini. Если сервер, к которому ошибочно подключается утилита, использует настройки по умолчанию для IP-адреса или порта. То требуется задать их вручную. Потому как, таким настройки меняются каждый раз, когда пользователь пытается подключиться, используя «Connect…».

2.4. Почему при запуске приложения я получаю сообщение следующего характера:



Просьба установить дополнительные библиотеки для работы приложения (http://www.microsoft.com/downloads/details.aspx?familyid=A5C84275-3B97-4AB7-A40D-3802B2AF5FC2&displaylang=en).

2.5. Почему при запуске приложения я получаю сообщение следующего характера:



Просьба установить дополнительные библиотеки для работы приложения (http://www.microsoft.com/downloads/details.aspx?familyid=A5C84275-3B97-4AB7-A40D-3802B2AF5FC2&displaylang=en). Ваши текущие библиотеки устарели.