УДК 004.42:004.771

Разработка комплекса программ мониторинга и управления рабочим столом удаленного компьютера

Буткина А. А., Кривова С. В., Шамаев А. В.

Национальный исследовательский Мордовский государственный университет

Аннотация: В статье описан процесс разработки комплекса программ мониторинга и управления рабочим столом удаленного компьютера, который может применяться при проведении научных мероприятий в дистанционной форме. В работе представлены результаты анализа сетевых протоколов, диаграммы вариантов использования, стек средств и технологий разработки, результаты тестирования работоспособности комплекса программ.

Ключевые слова: программное обеспечение, мониторинг, удаленное управление, клиент, сервер, сетевое взаимодействие.

В настоящее время в связи с возрастающей ролью дистанционных форм взаимодействия между участниками научного сообщества все большее значение приобретает организация возможности удаленного доступа организаторов научных мероприятий (международных конференций, школ-семинаров, форумов, защит диссертаций и т. п.) к компьютерам их участников. Поэтому разработка комплекса программ, позволяющего централизованно выполнять мониторинг и управление приложениями, установленными на компьютерах удаленных пользователей, является актуальной задачей. Целью данной работы является разработка подобного комплекса программ. Для достижения этой цели были поставлены и решены следующие задачи:

- изучение существующих технологий и протоколов удаленного доступа;
- анализ существующих аналогов разработанного комплекса программ;
- определение функциональных требований к разрабатываемому комплексу программ;
- выбор средств и технологий разработки;
- разработка архитектуры комплекса программ мониторинга и управления;
- реализация и тестирование разработанного комплекса программ.

Для организации удаленного управления используются различные варианты сетевых протоколов, среди которых наиболее широко распространены следующие:

- **Telnet** (telecommunications network) – сетевой протокол передачи данных телекоммуникационной сети;
- **SSH** (Secure Shell) – протокол прикладного уровня, обеспечивающий защищенное соединение;
- **RFB** (Remote Frame Buffer) – клиент-серверный сетевой протокол для удалённого доступа к графическому рабочему столу компьютера;
- **RDP** (Remote Desktop Protocol) – протокол, созданный компанией Microsoft для обеспечения удаленного доступа к серверам и рабочим станциям Windows.

В результате проведенного в работе анализа перечисленных сетевых протоколов для организации удаленного доступа в разработанном комплексе программ был использован протокол RDP. Рассмотрим принцип его работы более подробно. После установления соединения на транспортном уровне инициализируется удаленная сессия, в рамках которой согласуются различные параметры передачи данных [1]. После успешного завершения фазы инициализации сервер терминалов (удаленный пользователь) начинает передавать клиенту (наблюдателю) графический вывод и ожидает получение входных данных от устройств

ввода (клавиатуры и мыши). В качестве графического вывода может выступать как точная копия экрана, передаваемая как изображение, так и команды на отрисовку графических примитивов (прямоугольник, линия, эллипс, текст и другие). Передача вывода с помощью примитивов является приоритетной для протокола RDP, так как это позволяет значительно экономить трафик; а полное изображение передается лишь в том случае, если иное невозможно по каким-либо причинам. Удаленный клиент обрабатывает полученные команды и выводит изображения с помощью своей графической подсистемы. Сигналы нажатия и отпускания клавиши передаются отдельно с помощью специального флага. Введенные пользователем символы передаются путем отправки скан-кодов клавиатуры.

В настоящее время существует множество программных продуктов, реализующих функционал удаленного доступа. Все они отличаются используемыми протоколами, доступными функциями, системными требованиями и интерфейсами.

В работе был выполнен анализ следующего программного обеспечения, решающего задачу удаленного доступа:

- TeamViewer;
- Chrome Remote Desktop;
- Microsoft Remote Desktop;
- AnyDesk;
- AeroAdmin;
- Ammyy Admin;
- а также целый класс программ для удаленного мониторинга, которые не предоставляют возможности управления, а лишь позволяют просматривать удаленный рабочий стол (например, приложения для отслеживания деятельности офисных сотрудников).

Среди приведенных аналогов разрабатываемого комплекса программ наиболее широким функционалом обладает программный продукт TeamViewer [2], основными функциями которого являются:

- возможность осуществлять удаленное управление;
- возможность организовать совместную работу на одном устройстве;
- наличие файлового менеджера для организации обмена файлами;
- возможность обмена текстовыми сообщениями (функция «Чат»);
- возможность проведения онлайн-конференций в аудио- и видеоформате, а также в режиме демонстрации экрана (функция «Конференция»);
- возможность выполнения видеозаписи изображения экрана удаленного устройства.

Однако рассмотренные выше аналоги, во-первых, требуют приобретения дорогостоящих лицензий, во-вторых, имеют ограничения на количество одновременно подключенных устройств, в-третьих, содержат либо неполный, либо избыточный функционал, в связи с чем возникает потребность в создании нового программного продукта.

При проектировании комплекса программ был использован унифицированный язык моделирования UML, а для реализации программных модулей – платформа .NET и язык программирования C#, интерфейс прикладного программирования (API) Windows Desktop Sharing, средства создания интерфейса Windows Forms и объекты управления платформы ActiveX.

Модульная структура разрабатываемой системы с описанием ее интерфейса и всех ключевых функциональных элементов были ранее описаны авторами в [3].

Приведем описание функциональных требований к разработанной системе в виде диаграмм вариантов использования. Комплекс программ должен представлять собой два приложения для персонального компьютера, работающих под управлением операционной системы Microsoft Windows, программу-клиент и программу-наблюдатель. Программа-клиент должна предоставлять доступ к экрану компьютера, на котором она установлена. Она должна быть установлена на рабочие станции участников научных мероприятий и поз-

волять выполнить подключение к приложению-наблюдателю. Пользователь данной программы должен иметь возможность задавать настройки подключения, выбирать область демонстрации; выполнять подключение к программе-наблюдателю; производить обмен текстовыми сообщениями и файлами с организаторами научного мероприятия; регулировать доступ к управлению своим экраном, а также приостанавливать и возобновлять его демонстрацию. Программа-наблюдатель должна обнаруживать и обрабатывать имеющиеся клиентские подключения, выводить информацию о них и осуществлять доступ к удаленному компьютеру. Пользователями данной программы являются организаторы научных мероприятий, которые должны иметь возможность выполнять мониторинг или управление программным обеспечением компьютеров участников, а также обмениваться с ними файлами и текстовыми сообщениями.

Варианты использования для программы-клиента (Рис. 1) и для программы-наблюдателя (Рис. 2) выделены разными цветами в зависимости от типа реализуемого функционала:

- работа с подключением (синий цвет);
- мониторинг (желтый цвет);
- управление (зеленый цвет);
- общение (красный цвет).

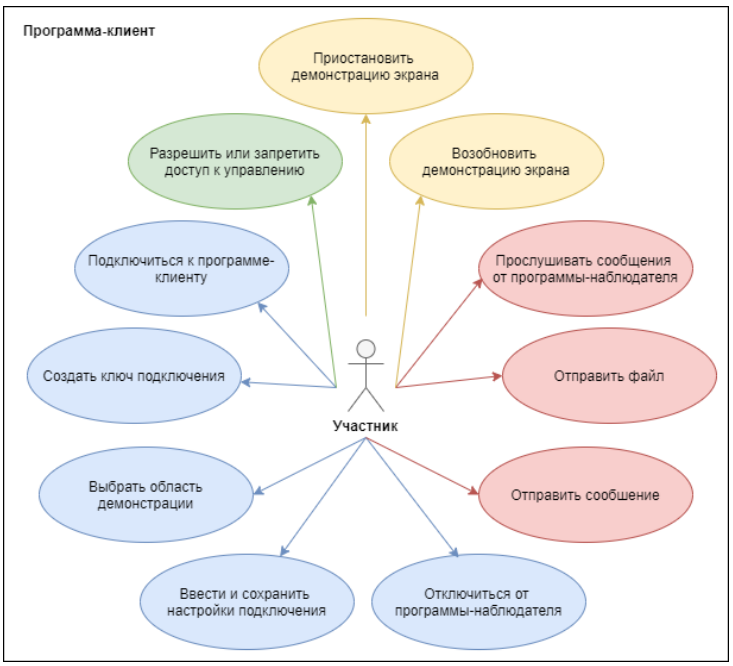

Рис. 1. Диаграмма вариантов использования для программы-клиента

На основе сформулированных функциональных требований к комплексу программ была разработана его логическая модель, представленная в виде совокупности диаграмм классов. Помимо диаграмм классов для программы-клиента и программы-наблюдателя, также потребовалось создание дополнительного решения **Utils**, содержащего классы, являющиеся общими для указанных программ.

Программа-клиент, диаграмма классов которой изображена на рис. 3, строится на основе трех классов: **FormClient**, **Client**, **Settings**, и ссылается на классы из общей библиотеки классов **Utils**.

Класс **FormClient** используется для создания экземпляра формы, которая отвечает за функционирование пользовательского интерфейса. Данный класс содержит поля ти-

Рис. 2. Диаграмма вариантов использования для программы-наблюдателя

пов Button (кнопка), TextBox (текстовое поле), RichTextBox (расширенное текстовое поле), CheckBox (флажок), CheckBoxList (список флажков). Объекты кнопок используются для выполнения команд пользователя, флажки – для выбора каких-либо опций, в текстовые поля вводится информация об участниках, их IP-адресах и номере порта сервера. Текстовые поля также используются для вывода информации: статуса операций, сгенерированной строки подключения, сообщений чата. Практически все методы в классе FormClient – обработчики событий манипулирования элементами управления формы. Кроме того, класс FormClient содержит поле client, которое отвечает за основную логику установления соединения с программой-наблюдателем и обработку данных.

Класс **Client** реализует интерфейс IDisposable, предназначенный для уничтожения неуправляемых ресурсов. В классе содержатся такие поля, как client (абстракция клиентского подключения по протоколу TCP), messageBrocker (отвечает за отправку и получение сообщений через сетевой поток NetworkStream), rdpSession (реализует сессию удаленного доступа), connectionString (хранит уникальную строку подключения), messageListener (поток, прослушивающий сообщения, поступающие от программы–наблюдателя), settings (задает настройки подключения), user (поле, описывающее пользователя), authorizationData (генерирует уникальную строку подключения), connect (флаг состояния подключения), listenOn (флаг статуса прослушивания сообщений). Также класс реализует следующие методы: ConnectTcp (подключение к серверу по протоколу TCP), CreateRdpSession (создает сессию удаленного доступа по протоколу RDP), CreateConnectionString (создает уникальную строку подключения на основании данных авторизации), Listening (прослушивает сообщения, поступающие от программы-наблюдателя, и выполняет различные действия в зависимости от типа сообщения), SendConnectionCommad (отправляет программе-наблюдателю команду о подключении клиента), Disconnect (отключение от программы-наблюдателя), ChangeControl (изменяет уровень доступа программы-наблюдателя к своему компьютеру), Incoming (добавляет в соответствующий список нового участника удаленной сессии), SetAppForSharing (перебирает список работающих приложений и устанавливает статус каждого из них), GetApplicationName (извлекает наименование окна программы).

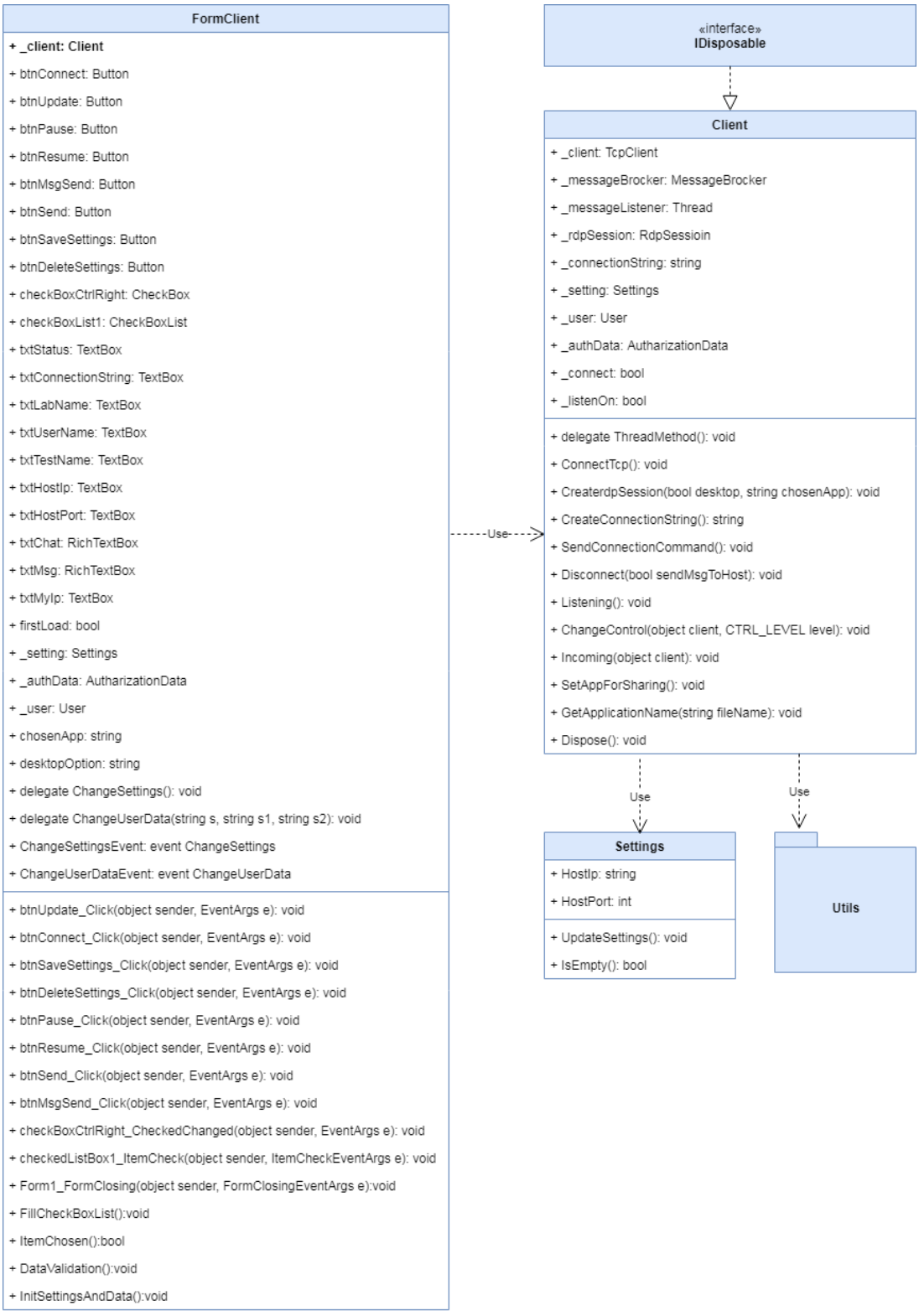

Рис. 3. Диаграмма классов для программы-клиента

Класс **Settings** содержит поля, обозначающие IP-адрес и номер порта сервера, а также методы для их проверки на пустоту и обновления.

На рис. 4 приведена диаграмма классов для пакета **Utils**, который содержит пакет моделей, пакет команд, класс **MessageBrocker**, реализующего функции обмена сообщениями, а также дополнительные классы **UpdateGUI** для обновления элементов управления форм и **CurrentIP** для вычисления собственного IP-адреса.

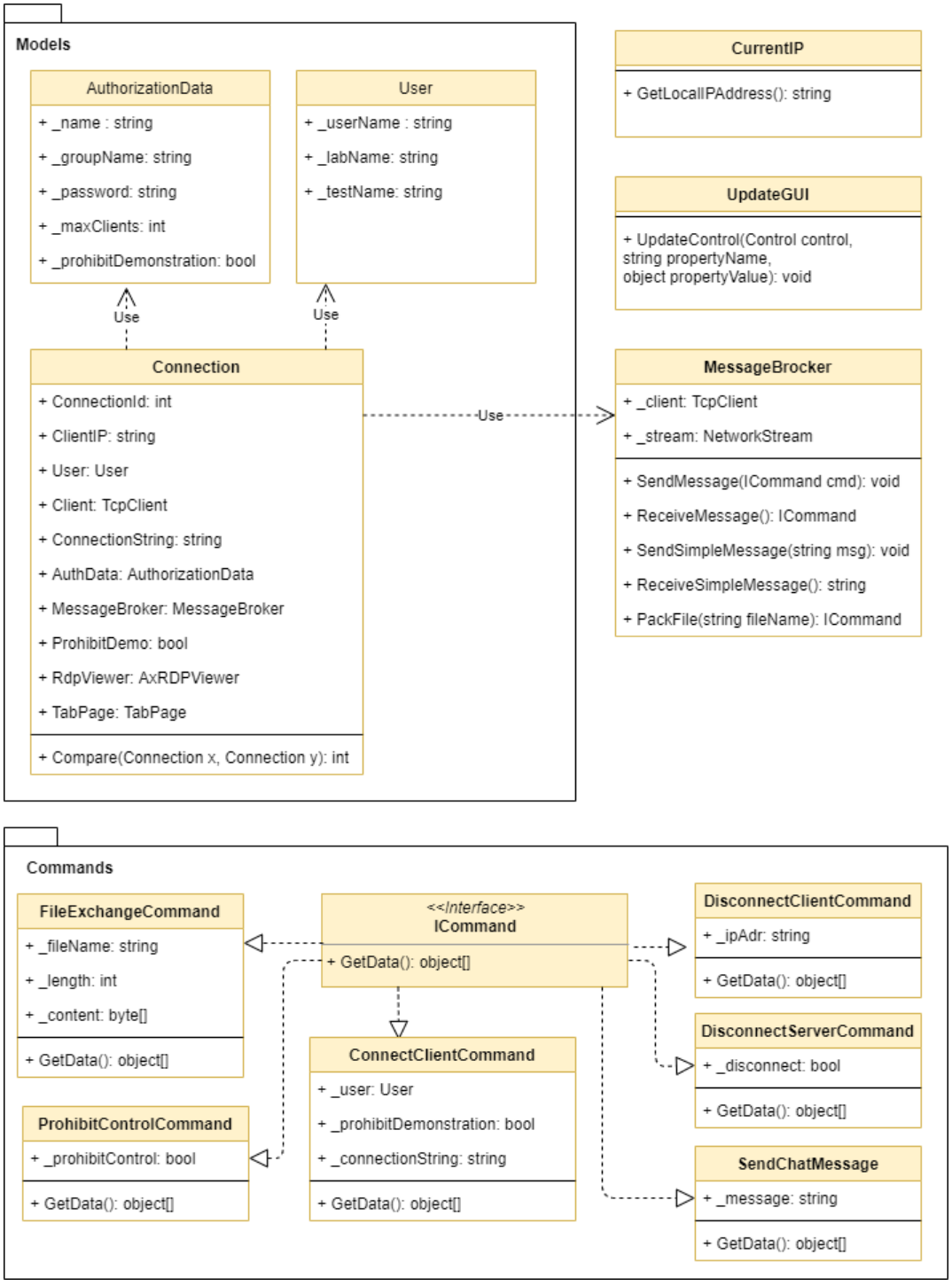

Рис. 4. Диаграмма классов для общей библиотеки классов Utils

Пакет моделей в **Utils** содержит класс **User**, который является абстракцией участника конференции (пользователя программы-клиента), класс **AuthorizationData**, который

содержит данные авторизации пользователя, необходимые для генерации уникальной строки подключения, и класс **Connection**, который является абстракцией клиентского подключения к серверу. Поля RdpViewer является экземпляром объекта управления платформы ActiveX, отвечающим за отображения экрана удаленного компьютера. Поле Tab – создаваемая для каждого нового клиентского подключения вкладка на форме программы-наблюдателя. Назначение остальных полей данного класса в пояснении не нуждается.

Пакет команд в библиотеке **Utils** содержит набор классов команд, пересылаемых между программой-клиентом и программой-наблюдателем, среди которых следует выделить интерфейс **ICommand**, который реализуют все остальные команды в пакете.

Диаграмма классов программы-наблюдателя (рис. 5) содержит класс **ViewerForm**, отвечающий за представление формы прослушивания подключений и логику обработки новых подключений, функций управления и демонстрации удаленного рабочего стола.

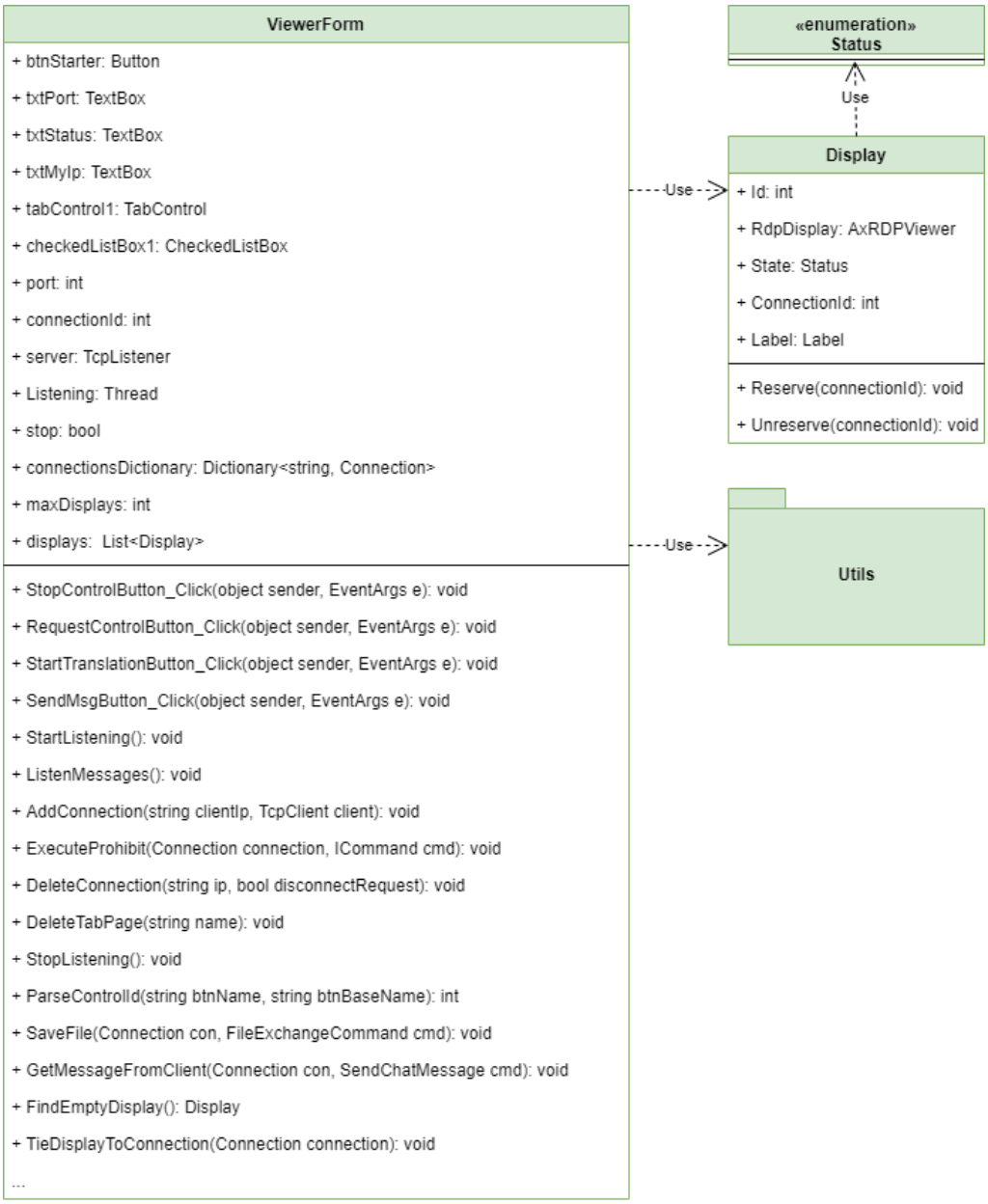

Рис. 5. Диаграмма классов для программы-наблюдателя

Класс ViewerForm использует класс **Display** – абстракцию экрана на вкладке программы-наблюдателя, который предназначен для отображения экрана рабочего стола удаленного клиента. Данный класс содержит метод Reserve для связывания экземпляра экрана с каким-либо из существующих подключений и метод Unreserve для удаления связи между экземпляром экрана и подключением. Класс **Display** в свою очередь использует перечисление Status, которое используется для отображения статуса экрана на вкладке программы-наблюдателя – занят (в данный момент транслирует экран удаленного пользователя) или свободен. Кроме того, класс **ViewerForm** также, как и класс **FormClient** ссылается на классы из общей библиотеки классов **Utils**.

В целях проверки работоспособности разработанного комплекса программ было выполнено тестирование всех вариантов его использования, представленных на рис. 1, 2, с применением соответствующих тестовых сценариев. Успешность выполнения всех рассмотренных авторами сценариев позволяет сделать вывод о корректности функционирования разработанного комплекса программ мониторинга и управления рабочим столом удаленного компьютера. Таким образом, данный комплекс программ может быть успешно применен для организации возможности удаленного доступа организаторов научных мероприятий к компьютерам их участников, что подтверждает практическую значимость работы.

Литература

1. Рублев С. Протокол Remote Desktop. Архитектура и возможности [Электронный ресурс]: SecurityLab.ru. – Режим доступа: https://www.securitylab.ru/analytics/367591.php.

2. Как работает TeamViewer: Полное руководство [Электронный ресурс]: TeamViewer. – Режим доступа: https://www.teamviewer.com/ru/документы/.

3. Кривова С. И., Шамаев А. В. Разработка программного обеспечения автоматизированной системы поддержки испытательного процесса // Материалы XXIII научно-практической конференции молодых ученых, аспирантов и студентов Национального исследовательского Мордовского государственного университета им. Н.П. Огарёва. сб. статей. – Саранск, 2019. – С. 235-238. – Режим доступа: https://www.elibrary.ru/download/elibrary_41178342_40846536.pdf.

MSC2020 68N19, 68M99

Development of a complex of programs for monitoring and managing a desktop of a remote computer

A. A. Butkina[1], S. I. Krivova[1], A.V. Shamaev[1]

National Research Ogarev Mordovia State University [1]

Abstract: The article describes the process of development complex of programs for monitoring and managing a desktop of a remote computer, which can be used when conducting scientific events in a remote form. The article presents the results of the analysis of network protocols, use case diagrams, a set of development tools and technologies, results of testing of the complex of programs.

Keywords: software, monitoring, remote control, client, server, networking.

References

1. Rublev S. Remote Desktop Protocol. Architecture and capabilities. SecurityLab.ru (in Russian). URL: https://www.securitylab.ru/analytics/367591.php (In Russian).

2. How TeamViewer Works: A Complete Guide. TeamViewer. URL: https://www.teamviewer.com/en/documents/ (In Russian).

3. Krivova S.I., Shamaev A.V. Development of the virtual simulator for test engineer of the laboratory of mechanical tests. Proceedings of the XXIII scientific and practical conference of young scientists, postgraduates and students of the National Research Ogarev Mordovia State University. Collection of articles. Saransk, 2019. pp. 235 238. URL: https://www.elibrary.ru/download/elibrary_41178342_40846536.pdf (in Russian).