DOI 10.37882/2223–2966.2022.10.27

СОЗДАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ МОНИТОРИНГА СЕРВЕРНОГО ОБОРУДОВАНИЯ

CREATION OF INFORMATION SYSTEM FOR MONITORING SERVER EQUIPMENT

B. Pruss
V. Romanov
O. Pleskacheva
V. Tsvetkov

Summary. The article describes the sequence of development of an information system (IS) for monitoring server equipment. The main problems that had to be faced in the design and development (IS) were identified. An analysis of analogue programs with consideration of their advantages and disadvantages was made. In the process of collecting and analyzing data on the subject area, the functional requirements put forward to the IS were identified. Based on them, a detailed design of the IC was made. The implementation, deployment and testing of the developed IS was carried out.

Keywords: information system, monitoring, server equipment.

Прусс Борис Наумович
Кандидат технических наук, доцент
Брянский государственный инженерно-технологический университет, г. Брянск
prussbor@gmail.com
Романов Виктор Александрович
Кандидат технических наук, доцент
Брянский государственный инженерно-технологический университет, г. Брянск
vromanov62@mail.ru
Плескачева Ольга Юрьевна
Кандидат педагогических наук, доцент
Брянский государственный Технический университет, г. Брянск
pleskacheva@inbox.ru
Цветков Владислав Владимирович
Магистрант
Брянский государственный инженерно-технологический университет, г. Брянск
hoooher32@yandex.ru

Аннотация. В статье описана последовательность разработки информационной системы (ИС) мониторинга серверного оборудования. Выявлены основные проблемы, с которыми пришлось столкнуться при проектировании и разработки (ИС). Произведен анализ программ-аналогов с рассмотрением их достоинств и недостатков. В процессе сбора и анализа данных о предметной области выявлены функциональные требования, выдвинутые к ИС. На их основе было произведено подробное проектирование ИС. Осуществлена реализация, развертывание и апробация разработанной ИС.

Ключевые слова: информационная система, мониторинг, серверное оборудование.

Мониторинг серверного оборудования в различных предприятиях является важной задачей, так как он необходим, в независимости от того, небольшая это компания или дата-центр. С помощью мониторинга сотрудники, ответственные за него, могут вовремя находить и устранять небольшие и критические неисправности, влияющие на все процессы, проходящие в компании.

При этом важными составляющими мониторинга должны быть не только своевременное оповещение, но и возможность анализа текущей работы оборудования, с целью предотвращения сбоев и хранения информации о них [1].

Следует отметить, что применение систем мониторинга позволяет решать следующие задачи:

1. Оптимизация использования информационных ресурсов;
2. Повышение качества и эффективности работы ИТ-подразделений, за счет скорости устранения сбоев в работе оборудования и программного обеспечения, а также минимизации времени их простоя;
3. Обеспечение надежности, безопасности и согласованного функционирования всех компонентов ИТ-инфраструктуры;
4. Облегчение модернизации ИТ-инфраструктуры.

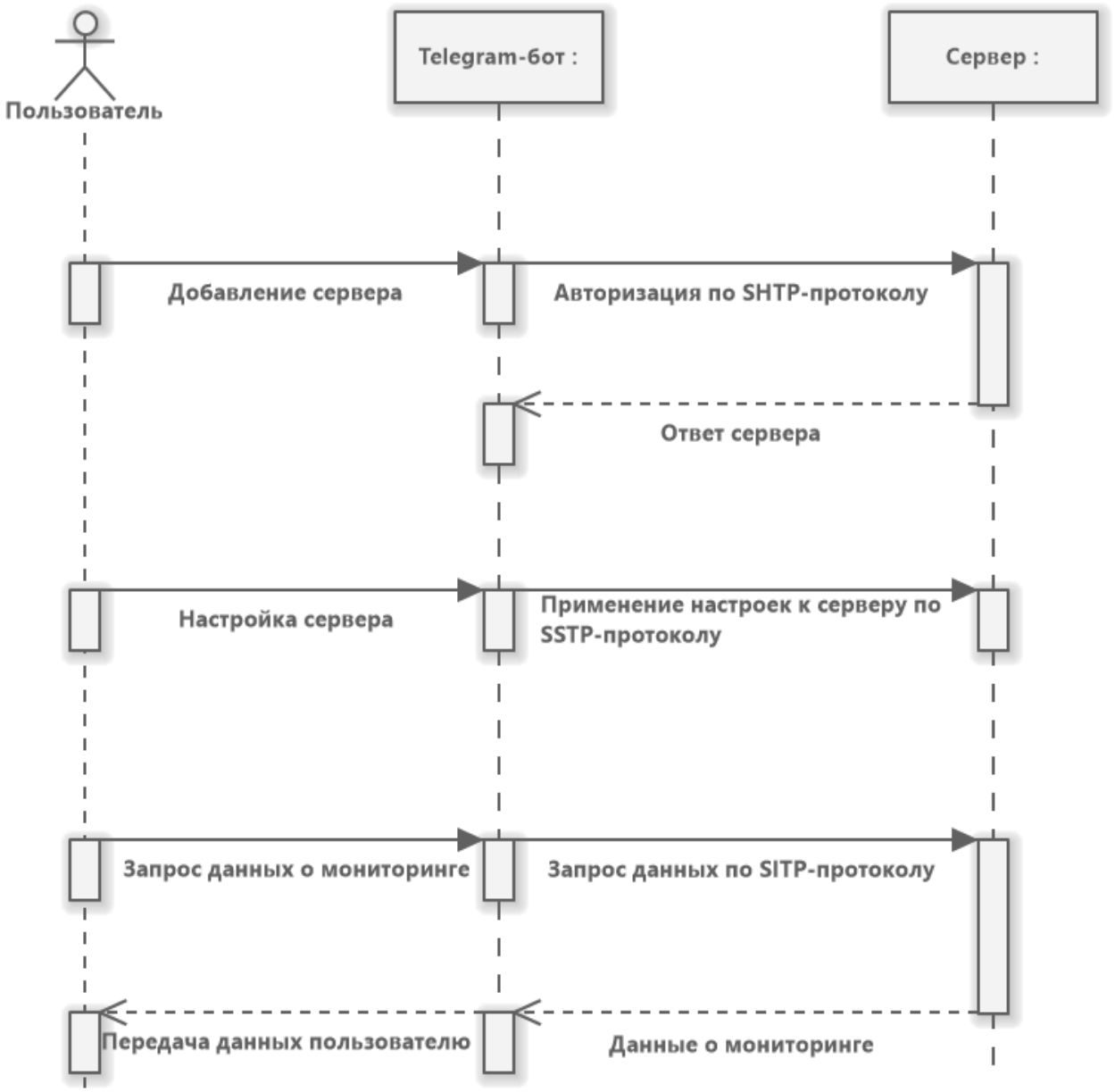

Рис. 1. Диаграмма последовательности

На сегодняшний при мониторинге оборудования сотрудники, отвечающие за это, сталкиваются со следующими проблемами:

1. Использование узконаправленного программного обеспечения;
2. Нечеткий мониторинг;
3. Отсутствие связи между пользователем и ИТ-подразделением.

Также следует отметить, что существующие программы-аналоги, такие как: Microsoft SCOM, Sematext Cloud, Datadog, несмотря на их широкий функционал, имеют ряд ограничений в использовании, таких как отсутствие кроссплатформенности и платная модель распространения.

На основании изложенных выше факторов и данных анализа достоинств и недостатков, приведенных выше программ-аналогов, было решено разработать концептуально новую информационную систему мониторинга серверного оборудования на платформе мессенджера «Telegram». Причем, данная информационная система должна быть проста в эксплуатации, а ее использование не должно требовать каких-либо специфических навыков или знаний.

Рис. 2. Схема меню Telegram-бота

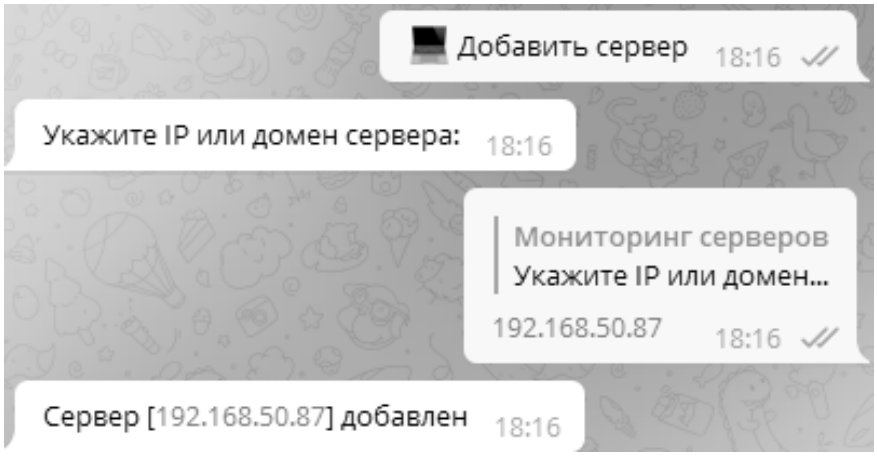

Рис. 3. Успешная валидация сервера

На основе проведенного анализа нами выделены следующие требования к информационной системе мониторинга:

1. Наличие возможности восстановления данных и требуемое состояние в случае прерывания или сбоя со стороны пользователя;
2. Реакция на все действия пользователя, работать и завершать работу в штатном режиме;
3. Бот не должен «падать», «зависать» или демонстрировать любое другое нестандартное поведение, в том числе при многократном быстром нажатии на какую-либо область дисплея, а также при нажатии на несколько областей дисплея одновременно (для мобильной версии мессенджера Telegram);
4. Обеспечение проверки корректности входных данных.

Для реализации указанных требований нами были спроектирована диаграмма последовательности разрабатываемой информационной системы, представленная на рис. 1.

Для реализации представленной диаграммы необходимо разработать следующие модули:

1. Клиентский модуль мониторинга;
2. Серверный модуль мониторинга;
3. Хостинг веб-приложений;
4. Сервер, реализующий протоколы TelegramAPI.

Ввиду того, что информационная система содержит в себе перечисленные модули, нами были выбраны следующие средства их разработки.

Для разработки клиентского модуля был выбран язык C#[2] и платформа.NET, а именно её реализация. NETCore версии 2.1. Так как данный модуль предназначен для сбора и пересылки информации с сервера, то он должен быть кроссплатформенным, так как серверы могут развертываться на различных операционных

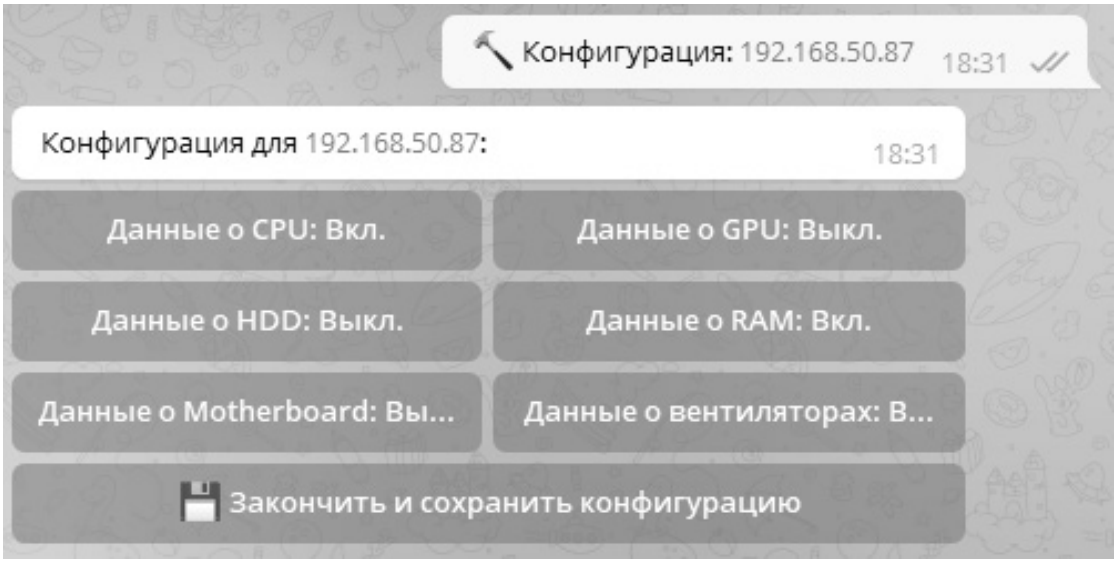

Рис. 4. Конфигурация сервера

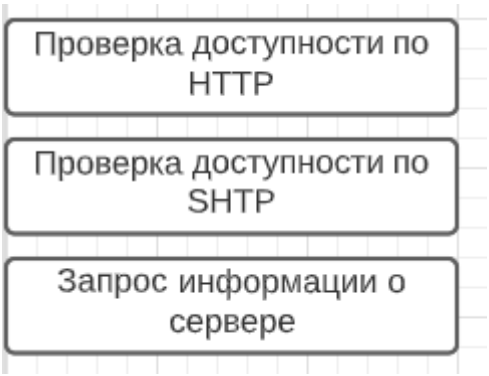

Рис. 5. Схема меню действий бота

системах. В дополнение следует отметить, что в нем должно быть реализовано наличие единого для всех платформ способа мониторинга оборудования, для чего была выбрана библиотека с открытым исходном OpenHardwareMonitor.

Серверный модуль ответственен за «общение» с Telegram. Для его разработки был выбран язык Python версии 3.8 [3] и его библиотека pyTelegramBotAPI версии 4.5.0 для разработки чат-бота, который должен иметь следующие функции:

1. Возможность добавления серверов для мониторинга;
2. Возможность конфигурации данных о мониторинге оборудования;
3. Возможность информирования пользователя о данных мониторинга серверного оборудования.

Интерфейс меню Telegram-бота [4] должен включать в себя возможности представленные на рис. 2.

Кнопка «Добавление сервера» предлагает пользователю ввести IP или домен сервера, требующего мониторинга оборудования. Далее введенные данные будут занесены в базу данных бота и в дальнейшем данный сервер будет ассоциироваться с данным пользователем Telegram (рис. 3).

Кнопка «Конфигурация сервера» предложит пользователю выбрать один из ассоциируемых с ним серверов, далее появится меню конфигурации для данного сервера (рис. 4).

Кнопка «Мониторинг сервера» вновь предложит пользователю выбрать один из ассоциируемых с ним серверов. После выбора нужного сервера должно воз-

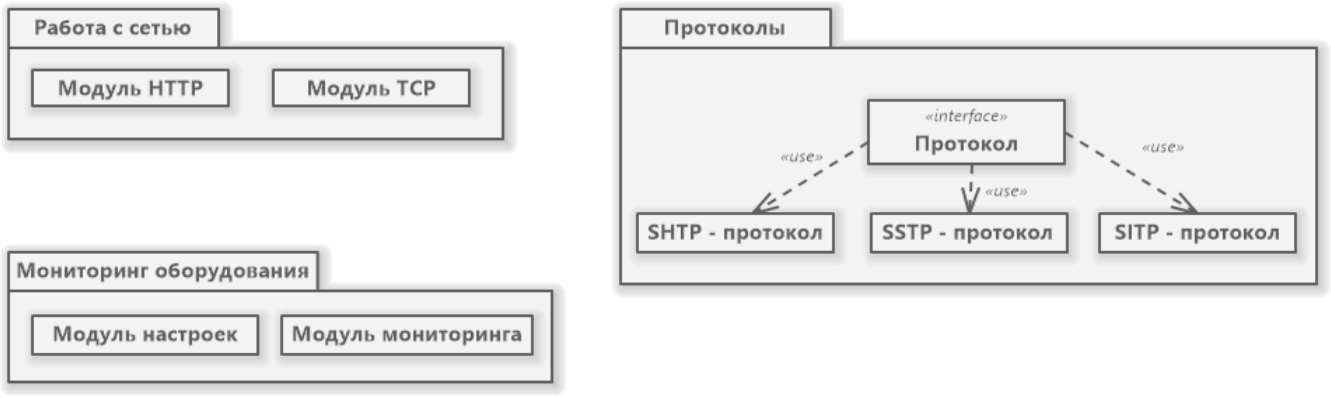

Рис. 6. Архитектура клиентского модуля

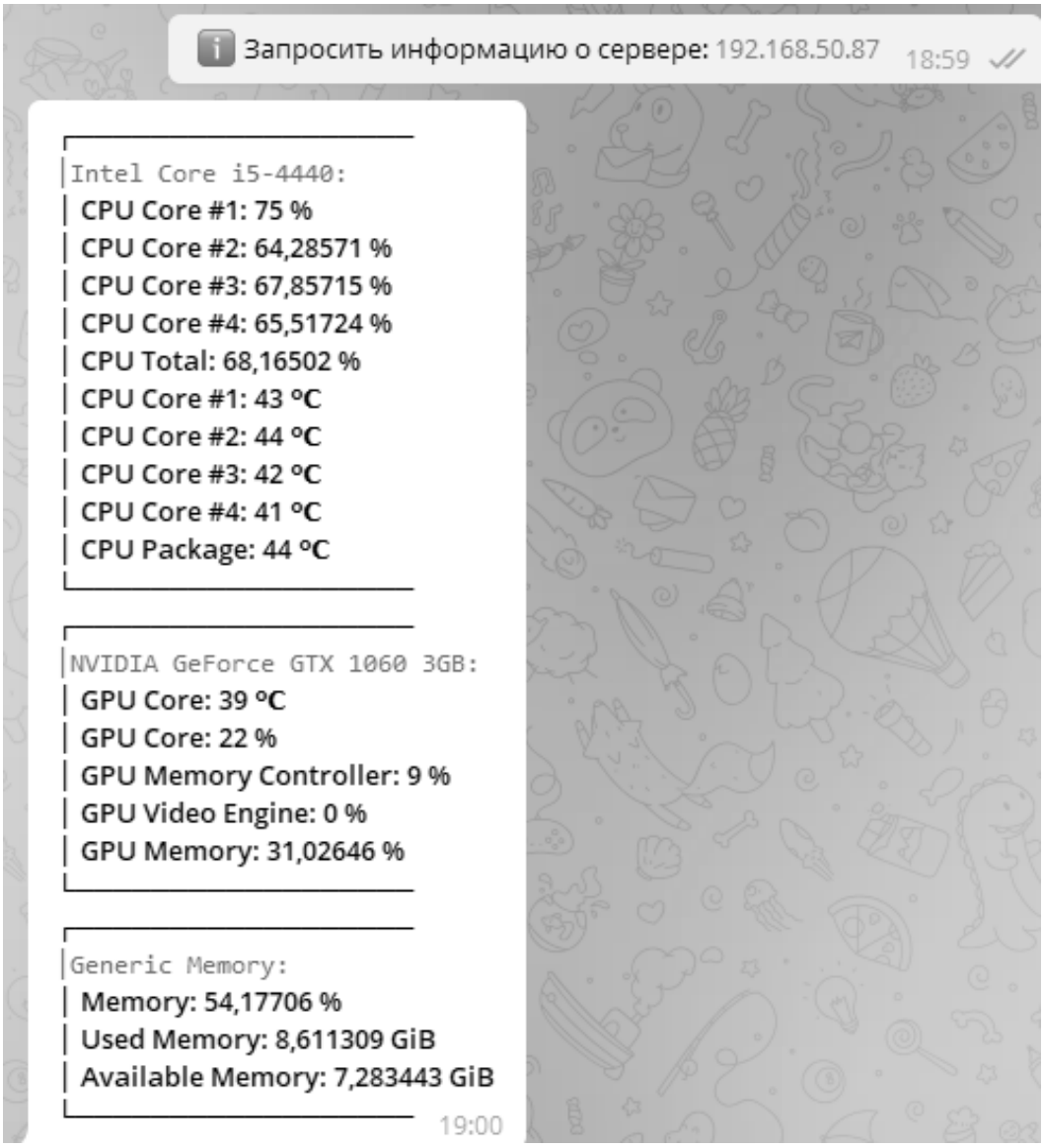

Рис. 7. Результат работы системы

никнуть еще одно меню, предлагающее три действия на выбор (рис. 5).

Пункт меню «Проверка доступности по HTTP» будет посылать серверу пустой HTTP-запрос, если код ответа будет равен 200, выводить положительный результат. Будет использоваться для проверки возможности обмена данными между ботом и сервером.

Следующий пункт «Проверка доступности по SHTP» будет посылать серверу SHTP — запрос, сервер должен будет ответить на него и тогда бот выведет положительный результат. Будет использовать для проверки работоспособности клиентского модуля.

Кнопка «Запрос информации о сервере» реализовывает главную функцию бота, а именно отправку SITP — запроса на сервер, приём и форматирование данных об оборудовании сервера.

Для инициализации Telegram-бота необходимо получить токен у официального бота Telegram «BotFather», который, как следует из названия является «отцом» всех ботов в Telegram.

Рассматривая клиентский модуль необходимо выделить следующие требования, предъявляемые к нему:

1. Кроссплатформенность;
2. Поддержка протоколов SHTP, SSTP, SITP;
3. Поддержка HTTP и TCP соединений;
4. Возможность конфигурирования.

Клиентский модуль было решено разработать на платформе.NET, точнее на её реализации.NETCore. В качестве среды разработки была выбрана VisualStudio 2017.

Выбор.NETCore обусловлен несколькими факторами:

1. 1.Кроссплатформенность;
2. 2.Отсутствие зависимостей при установке на сервер;
3. 3.Наличие единого для всех платформ способа мониторинга оборудования.

Если первый пункт в пояснениях не нуждается, то остальные два разберём поподробнее:

- отсутствие зависимостей при установке на сервер. В VisualStudio для проектов.NETCore имеется возможность выгрузки ядра, всех зависимостей и библиотек, что позволяет не устанавливать саму среду выполнения dotnet на целевую электронно-вычислительную машину;
- наличие единого для всех платформ способа мониторинга оборудования. Для мониторинга состояния серверного оборудования была выбрана библиотека с открытым исходном OpenHardwareMonitor.

Архитектура клиентского модуля представлена на рис. 6.

В качестве хостинга для серверного модуля был выбран облачный сервис PythonAnywhere. Его выбор обусловлен тем, что он соответствует всем нашим требованиям, а именно:

1. Возможность развёртывания приложений на Python;
2. Поддержка MySQL;
3. Веб-редактор кода;
4. Наличие SSL-сертификата для HTTPS-соединения.

Важным аспектом при разработке ИС является передача данных. Специально для данной ИС было разработано три протокола обмена данных между сервером, подвергающимся мониторингу и принимающей стороной, в данном случае являющейся сервером Telegram-бота:

1. ServerHelloTransferProtocol (SHTP) — так называемый «приветственный» протокол. Используется для валидации принадлежности сервера. Отправитель формирует пакет с текстом «hello» в кодировке UTF-8 и посылает принимающей стороне. Принимающая сторона должна отправить в ответ точно такую же строку, после этого валидация будет пройдена;
2. ServerSettingsTransferProtocol (SSTP) — протокол передачи данных о конфигурации мониторинга, предназначен для настроек, имеющих два значения: вкл. и выкл. Хранит каждую настройку как отдельный бит числа, начиная с младшего;
3. ServerInfoTransferProtocol (SITP) — протокол передачи данных, собранных в процессе мониторинга, и передающий их в виде древовидной структуры.

Представленные выше протоколы данных могут использовать как протокол транспортного уровня TCP, так и протокол прикладного уровня HTTP для межсетевой транспортировки. Результаты работы системы представлены на рис. 7.

Подводя итоги, можно сказать, что разработанная ИС соответствует поставленной цели. Она позволяет осуществлять мониторинг серверного оборудования по заданным параметрам, что повышает качество работы ИТ-подразделений компании, за счет оперативных реакций на внештатные ситуации, а также повышает качество работы всей организации. Отдельно хоте-

лось бы выделить тот факт, что мониторинг осуществляется по многим параметрам и всей номенклатуре серверного оборудования, независимо от аппаратных и программных характеристик. При этом установка и развертывание системы не требует больших аппаратных и человеческих ресурсов.

ЛИТЕРАТУРА

1. Кузин, А.В. Компьютерные сети: учебное пособие / А.В. Кузин, Д.А. Кузин. — М.: Форум, 2018. — 704 с.
2. Тюкачев Н.А. C#. Основы программирования: учебное пособие для вузов / Н.А. Тюкачев, В.Г. Хлебостроев. — 4-е изд., стер. -СПб.: Лань, 2021. — 272 с.
3. Северанс Ч.Р. Python для всех / Ч.Р. Северанс; перевод с английского А.В. Снастина. — М.: ДМК Пресс, 2022. — 262 с.
4. Классификация и методы создания чат-бот приложений [Электронный ресурс]. — URL: https://cyberleninka.ru/article/n/ klassifikatsiya-i-metody-sozdaniya-chatbot-prilozheniy/viewer(Дата обращения: 18.07.2022).

г. Брянск