ORES
Научно-практический электронный журнал
Оригинальные исследования (ОРИС), *ISSN 2222-0402*
www.ores.su
original-research.ru

ВОЗМОЖНОСТИ СТАНДАРТИЗАЦИИ ПРОТОКОЛА ДЛЯ СЕВЕРНОГО ИНТЕРФЕЙСА В ПРОГРАММНО-РАЗДЕЛЯЕМЫХ СЕТЯХ

Крылова Дарья Николаевна,
Студент группы ИУК5-61Б Калужского филиала федерального государственного автономного образовательного учреждения высшего образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)», 248000, Россия, г. Калуга, ул. Баженова, д.2
kryylovadn@student.bmstu.ru

Федоров Виктор Олегович,
Кандидат технических наук, доцент Калужского филиала федерального государственного автономного образовательного учреждения высшего образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)», 248000, Россия, г. Калуга, ул. Баженова, д.2.
fedorov_vo@bmstu.ru

Аннотация

Программно-конфигурируемые сети (SDN) являются перспективным решением сетевых проблем устройств Интернета вещей (IoT). Их ключевая особенность заключается в разделении плоскостей управления и передачи данных, что позволяет централизовать сетевую логику на программном контроллере. Такой подход упрощает администрирование сети, повышает эффективность использования ресурсов, обеспечивает гибкость управления и снижает сложность эксплуатации. Взаимодействие контроллера с сетевыми устройствами осуществляется через протокол OpenFlow, ставший стандартом южного интерфейса SDN. Однако для северного интерфейса, обеспечивающего интеграцию приложений и обмен данными между компонентами системы, единый стандарт пока отсутствует. С ростом потребности в распределённых и периферийных вычислениях возрастает необходимость в стандартизации протоколов северного интерфейса. В статье рассматриваются существующие решения для его реализации, проводится их сравнительный анализ и оцениваются возможности интеграции в приложения IoT.

Ключевые слова: программно-разделяемые сети (SDN), северный интерфейс (Northbound Interface, NBI), стандартизация протоколов, контроллер SDN, интернет вещей (IoT), REST API.

PROTOCOL STANDARDIZATION CAPABILITIES FOR THE NORTHBOUND INTERFACE IN SOFTWARE-DEFINED NETWORKS

Krylova Daria Nikolaevna,
Student of the group IUK5-61B of the Kaluga branch of the Federal State Autonomous Educational Institution of Higher Education "Bauman Moscow State Technical University (National Research University)", 248000, Russia, Kaluga, Bazhenova str., 2
kryylovadn@student.bmstu.ru

Fedorov Viktor Olegovich,
Candidate of Technical Sciences, Associate Professor at the Kaluga Branch of the Federal State Autonomous Educational Institution of Higher Education "Bauman Moscow State Technical University (National Research University)", 248000, Russia, Kaluga, Bazhenova Street, 2.
fedorov_vo@bmstu.ru

ABSTRACT

Software-configurable networks (SDNs) are a promising solution to the network problems of Internet of Things (IoT) devices. Their key feature is the separation of the control and data transmission planes, which makes it possible to centralize network logic on a software controller. This approach simplifies network administration, increases resource efficiency, provides management flexibility, and reduces operational complexity. The controller interacts with network devices via the OpenFlow protocol, which has become the standard for the southern SDN interface. However, there is still no single standard for the northern interface, which provides application integration and data exchange between system components. With the growing demand for distributed and peripheral computing, the need for standardization of the Northern interface protocols is increasing. The article examines the existing solutions for its implementation, conducts their comparative analysis and evaluates the possibilities of integration into IoT applications.

Keywords: software-shared networks (SDN), Northbound Interface (NBI), protocol standardization, SDN controller, Internet of Things (IoT), REST API.

Введение

В последние годы развитие облачных вычислений, технологий Интернета вещей и виртуализации привело к усложнению сетевой инфраструктуры. Современные сети связи обеспечивают не только передачу данных, но и поддержку функций управления, автоматизации и мониторинга. Ключевым элементом архитектуры является северный интерфейс (NBI) - программный мост между контроллером и приложениями.

Отсутствие единого стандарта NBI приводит к тому, что каждая платформа предлагает собственный API с разными форматами данных и принципами аутентификации. Это ведёт к снижению совместимости систем и росту трудозатрат на интеграцию.

Концепция программно-определяемых сетей (SDN) разделяет плоскости данных и управления, перенося функции контроля в централизованные контроллеры. Это породило необходимость в чётко определённом северном интерфейсе для взаимодействия с бизнес-приложениями.

Единый подход к проектированию NBI до сих пор не сформирован. Разнообразие протоколов (REST, gRPC, NETCONF) и форматов данных (JSON, XML, YAML) создаёт технологическую мозаику, где компоненты не складываются в целостную систему [1, 2].

Теоретические основы северного интерфейса и его роль в архитектуре систем

Формирование понятия северного интерфейса неразрывно связано с развитием программно-определяемых сетей (SDN). Традиционные сети, состоявшие из собственных (закрытых) физических устройств разных производителей, демонстрировали негибкость, сложность управления и препятствия для внедрения новых услуг. Ответом стала SDN, главный принцип которой - разделение плоскости управления и плоскости передачи данных. Это позволило перенести функции управления в централизованный контроллер.

Южный интерфейс отвечает за связь с сетевым оборудованием, а северный - за связь с прикладным уровнем. Термин «северный интерфейс» стал обозначать программный интерфейс, через который приложения общаются с контроллером, узнают состояние сети и меняют её настройки [2, 3].

Северный интерфейс — это API, обеспечивающий взаимодействие между контроллером SDN и приложениями [1]. Его главные задачи: упрощение (предоставление единого представления сетевых ресурсов, скрывая различия оборудования), управление (запрос состояния сети и изменение её работы) и автоматизация (обеспечение качества связи, быстрое восстановление после сбоев, объединение ресурсов) [4].

Единой классификации северных интерфейсов не существует. По типу протоколов чаще всего используется RESTful API, также применяются NETCONF/RESTCONF с языком YANG, собственные интерфейсы под конкретную платформу и gRPC. Главная проблема - жёсткая привязка к определённому контроллеру (OpenDaylight, ONOS, Floodlight), что затрудняет перенос приложений между разными системами. По области применения северный интерфейс используется в транспортных сетях, облачных системах управления (Kubernetes, OpenStack), системах высокой надёжности и в интернете вещей [5].

Проектирование северного интерфейса требует соблюдения ряда правил. Масштабируемость и надёжность: контроллер не должен быть единственной точкой отказа [6]. Безопасность: необходимы проверка подлинности, разграничение прав доступа и шифрование данных. Единообразие описания данных: наиболее подходящие способы - YANG, JSON Schema, OpenAPI. Также важны полное описание и проверка работы интерфейса.

Практические подходы и опыт создания северных интерфейсов

Реализация северных интерфейсов в разных SDN-контроллерах показывает остроту проблемы разнородности. Контроллеры с открытым кодом (OpenDaylight, ONOS) предлагают мощные, но часто несовместимые интерфейсы на основе RESTCONF и REST API [1, 2]. Контроллер Floodlight использует двойную модель: Java API для встраивания логики в контроллер и REST API для связи с внешними системами [5]. Закрытые решения Cisco (ACI, DNA Center) используют RESTful API и NETCONF, хотя их подход отличается от «чистой» SDN из-за сохранения элементов распределённого управления на устройствах.

В облачных системах северный интерфейс становится ключевым элементом для предоставления сетевых услуг по требованию. Транспортные SDN (Huawei, Netcracker) используют REST API для связи с системами управления облачными ресурсами, что позволяет реализовать, например, предоставление пропускной способности по требованию между центрами обработки данных.

Анализ практического опыта выявляет ряд проблем. Во-первых, несогласованность форматов данных и терминологии: разные контроллеры используют свои модели данных и названия, что делает невозможным создание универсальных приложений [3]. Это связано с разными требованиями приложений и борьбой производителей. Во-вторых, сложности при объединении разнородных платформ, особенно в крупных сетях, где работают устройства разных поколений и производителей. В-третьих, проблемы обновления и совместимости: обновление контроллера может нарушить работу приложений, а протокол NETCONF из-за фазы подтверждения вносит заметную задержку, что неприемлемо для задач с восстановлением за миллисекунды.

Применение северных интерфейсов в IoT требует продуманных решений. REST остаётся самым распространённым вариантом благодаря простоте и удобству связи с веб-технологиями [8]. Пример - система умного дома, где микроконтроллер работает как веб-сервер, а приложение отправляет команды через REST-вызовы. Использование JSON вместо XML даёт более высокую скорость [7].

Главная проблема IoT - различие устройств по мощности, памяти и питанию. Для её решения предложена гибридная архитектура с прямым и косвенным управлением через умный шлюз. Прямое управление подходит для мощных устройств (камеры, датчики безопасности) - оно даёт минимальную задержку. Косвенное управление - для слабых устройств на батарейках: шлюз работает посредником, экономя их ресурсы. Это позволяет системе работать со всеми типами устройств, обеспечивая низкую задержку и энергоэффективность [9].

Для систем, где нужны высокая скорость, минимальные задержки и потоковая передача данных (телеметрия, мониторинг в реальном времени), лучше подходит gRPC. Его преимущества - двоичный протокол Protocol Buffers и поддержка потоковых вызовов. Испытания показывают, что gRPC быстрее при передаче больших объёмов данных, а в условиях нестабильной связи (3G) сокращает время ответа на 25% по сравнению с REST [10].

IoT-системы тесно связаны с мобильными приложениями. Для ускорения разработки используют готовые серверные решения BaaS (Firebase, AWS Amplify). Для разгрузки мобильных устройств применяют передачу сложных задач в облако, но её эффективность сильно зависит от качества сети: при задержке более 100 мс производительность может падать на 30-40%.

Соединение облачных технологий с IoT создаёт риски для безопасности: различия в настройках защиты на разных устройствах Android, ограниченный контроль над шифрованием в BaaS, риск перехвата данных. Для снижения уязвимостей предлагают смешанное шифрование (AES-256 локально и гомоморфное шифрование в облаке), что снижает число уязвимостей до 40%.

Таким образом, универсального решения для IoT не существует. REST подходит для простых сценариев с небольшим объёмом данных. gRPC - для нагруженных систем и мониторинга в реальном времени. Гибридная архитектура позволяет объединять разные устройства, а интеграция с мобильными и облачными платформами требует учёта качества сети и мер безопасности.

Сравнение SDN-контроллеров показывает: OpenDaylight имеет наиболее широкий набор интерфейсов. Но с точки зрения разработчика приложений главное - гибкость API и независимость от платформы, и здесь ситуация неудовлетворительная. Решения на основе REST лучше по открытости и простоте, а NETCONF/YANG дают более строгую логику и повышенную безопасность. Экосистема северных интерфейсов пока недостаточно зрелая, но у неё большой потенциал для стандартизации.

Стандартизация северного интерфейса: концепции, модели и прогнозы

Ведущие международные организации осознали необходимость стандартизации северного интерфейса. Open Networking Foundation (ONF), которая уже успешно стандартизировала южный интерфейс OpenFlow, создала специальную рабочую группу Northbound для разработки прототипов и формирования отраслевого стандарта NBI. Другие организации, такие как IETF, вносят вклад через разработку спецификаций RESTCONF, NETCONF и языка моделирования данных YANG [7]. Параллельно ведутся работы по стандартизации Transport API (TAPI) для транспортных сетей.

В основе современной концепции стандартизации лежит идея создания универсального, модельно-ориентированного API. Он должен быть модульным (чёткое разделение функций на независимые компоненты, например, отдельно для мониторинга, управления потоками и политиками безопасности), расширяемым (возможность добавлять новые функции без нарушения совместимости с существующими приложениями) и прозрачным (чёткое и понятное описание всех операций). Предлагаемая модель часто имеет слоевую структуру, включающую модули для приёма уведомлений о событиях, для анализа пакетов и для управления потоками. В качестве основных форматов описания

данных рассматриваются YANG (для моделирования конфигурации сети), OpenAPI (для описания RESTful API) и JSON-LD (для представления связных данных).

Переход к единому стандарту не может быть мгновенным, так как это нарушит работу существующих инфраструктур. Наиболее практичный подход - поэтапное внедрение через промежуточный программный слой (shim layer), который будет преобразовывать вызовы нового стандартного NBI в команды проприетарных интерфейсов уже работающих контроллеров [2]. Это обеспечит плавный переход и обратную совместимость. Стандартизация даст разработчикам возможность писать приложения один раз и запускать их на любой совместимой платформе, что снизит стоимость владения и ускорит вывод продуктов на рынок. Для операторов связи это означает снижение зависимости от конкретного производителя и упрощение работы с разнородным оборудованием.

Сравнительный анализ протоколов REST, NETCONF и gRPC представлен в таблице 1 (основные особенности, преимущества и недостатки) и в таблице 2 (сравнение по критериям производительности, безопасности, удобству разработки, поддерживаемым языкам и области применения).

Таблица 1 - Сравнительный анализ протоколов

Название протокола	Основные особенности	Преимущества	Недостатки
REST	Основан на HTTP/HTTPS, прост в разработке, широко используется для открытых API, поддерживает GET/POST/PUT/DELETE	Лёгкая интеграция с веб-сервисами, низкие требования к клиентам, хорошая масштабируемость	Плохо подходит для больших объёмов данных, падение производительности при большом числе запросов
NETCONF	Специализированный протокол для управления сетевыми устройствами, использует XML и RPC, поддерживает транзакции (commit/rollback)	Высокая надёжность и безопасность, контроль операций через транзакции, подходит для устаревших устройств	Сложнее REST в разработке, требует много вычислительных ресурсов
gRPC	Создан Google для высокой производительности, работает поверх TCP/IP, поддерживает бинарные данные, использует Protocol Buffers	Высокая скорость даже при больших объёмах, компактность сообщений, совместимость с разными языками программирования	Менее интуитивен и сложнее REST, требует настройки клиента и сервера

Таблица 2 - Сравнительный анализ протоколов по критериям

Критерий	REST	NETCONF	gRPC
Производительность	Средняя	Низкая	Высокая
Безопасность	Хорошая	Отличная	Хорошая
Удобство разработки	Очень хорошее	Среднее	Среднее

Поддерживаемые языки	Широкий спектр	Только поддерживающие XML	Большинство распространенных языков
Применение	Универсальное, преимущественно веб-сервисы	Конфигурация и управление сетевыми устройствами	Интерактивные высокоэффективные системы

Выбор подходящего протокола зависит от конкретных задач. Если важны простота и широкая поддержка, выбирается REST. Если требуются высокий уровень безопасности и работа с устаревшими сетевыми устройствами, предпочтение отдаётся NETCONF. Для высоконагруженных систем с большими объёмами данных лучше всего подходит gRPC.

Будущее развитие северного интерфейса видится в нескольких направлениях. Во-первых, NBI превратится из канала простого управления в основной канал для интеллектуальной автоматизации: самовосстановление сети, динамическое перераспределение ресурсов в реальном времени и гарантированное выполнение соглашений об уровне обслуживания [4, 6]. Во-вторых, стандартизированный API станет основой для интеграции с системами искусственного интеллекта и машинного обучения, что позволит прогнозировать сетевые аномалии, проводить предиктивное обслуживание и перейти к когнитивному управлению сетью. В-третьих, движение в сторону стандартизации TAPI и развитие мультидоменных архитектур указывают на долгосрочную цель - создание единого глобального интерфейса для управления разнородными сетями будущего.

Вывод

Северный интерфейс является не просто техническим компонентом, а фундаментальной архитектурной концепцией, лежащей в основе гибкости и автоматизации современных сетей. Несмотря на его критически важную роль в SDN и облачных вычислениях, отрасль сталкивается с серьёзной проблемой - отсутствием единого стандарта. Это ведёт к фрагментации экосистемы, технологическим барьерам и росту затрат на интеграцию.

Теоретические основы северного интерфейса хорошо проработаны: его роль как уровня абстракции, функциональные задачи и требования к проектированию чётко описаны в научных публикациях. Практическая реализация в SDN-контроллерах и облачных платформах показывает огромный потенциал технологии, но также обнажает проблемы, вызванные отсутствием согласованности и технологическим разобщением.

Преодоление этих проблем связано с работой международных организаций по стандартизации и исследованиями в области унифицированного модельно-ориентированного API. Успех в этом направлении откроет путь к интеллектуальной и автономной автоматизации сетей. Северный интерфейс станет ключевым элементом, обеспечивающим интеграцию бизнес-приложений с сетевыми ресурсами. Дальнейшие исследования должны быть сосредоточены на разработке формальных моделей данных, эталонных реализаций и методик перехода от закрытых решений к открытым стандартам.

Список литературы:

1. Агеева, А. Д. Транспортные программно-конфигурируемые сети / А. Д. Агеева, Н. В. Бирюкова, В. В. Мошков, В. С. Елагин // Modern Science. – 2019. – № 12-4. – С. 291–301.
2. Алексеева, К. Ю. Анализ работы интерфейсов в программно-конфигурируемых сетях / К. Ю. Алексеева // Теория и практика современной науки. – 2020. – № 5 (59). – С. 513–516.

3. Намиот, Д. Е. Интерфейсы прикладного уровня в SDN / Д. Е. Намиот // Современные информационные технологии и ИТ-образование. – 2015. – Т. 2, № 11. – С. 26–30.
4. Семеновых, А. А. Сравнительный анализ SDN-контроллеров / А. А. Семеновых, О. Р. Лапонина // International Journal of Open Information Technologies. – 2018. – Т. 6, № 7. – С. 50–56.
5. Alghamdi, A. Designing a RESTful Northbound Interface for Incompatible Software Defined Network Controllers / A. Alghamdi, D. Paul, E. Sadgrove // SN Computer Science. – 2022. – Vol. 3. – Art. 502. – DOI 10.1007/s42979-022-01423-3.
6. Sahoo, K. S. Software Defined Network: The Next Generation Internet Technology / K. S. Sahoo, S. K. Mishra, S. Sahoo, B. Sahoo // International Journal of Wireless and Microwave Technologies. – 2017. – Vol. 7, No. 2. – pp. 13–24. – DOI 10.5815/ijwmt.2017.02.02.
7. Zaw Lin Oo. IoT Based Home Automation System using a REST API Architecture / Zaw Lin Oo, Theint Win Lai, Aung Moe // European Journal of Technique. – 2022. – Vol. 12, No. 2. – pp. 123–128. – DOI 10.36222/ejt.1018131.
8. Maurya, R. Application of Restful APIs in IOT: A Review / R. Maurya, K. A. Nambiar, P. Babbe, J. P. Kalokhe, Y. S. Ingle, N. F. Shaikh // International Journal for Research in Applied Science & Engineering Technology. – 2021. – Vol. 9, Issue II. – pp. 145–151. – DOI 10.22214/ijraset.2021.33013.
9. Pham, C. Management Architecture for Heterogeneous IoT Devices in Home Network / C. Pham, Y. Lim, Y. Tan // 2016 IEEE 5th Global Conference on Consumer Electronics (GCCE). – 2016. – DOI 10.1109/GCCE.2016.7800448.
10. Buzhin, I. G. Comparative Analysis of the REST and gRPC Used in the Monitoring System of Communication Network Virtualized Infrastructure / I. G. Buzhin, A. Yu. Derevyankin, V. M. Antonova, A. P. Perevalov, Yu. B. Mironov // T-Comm: Телекоммуникации и транспорт. – 2023. – Т. 17, № 4. – С. 50–55. – DOI 10.36724/2072-8735-2023-17-4-50-55.

References:

1. Ageeva, A.D. Software-configurable transport networks / A.D. Ageeva, N. V. Biryukova, V. V. Moshkov, V. S. Elagin // Modern Science. – 2019. – No. 12-4. – pp. 291-301.
2. Alekseeva, K. Y. Analysis of interfaces in software-configurable networks / K. Y. Alekseeva // Theory and practice of modern science. – 2020. – No. 5 (59). – pp. 513-516.
3. Namiot, D. E. Interfaces of the application layer in SDN / D. E. Namiot // Modern information technologies and IT education. – 2015. – Vol. 2, No. 11. – pp. 26-30.
4. Semenovykh, A. A. Comparative analysis of SDN controllers / A. A. Semenovykh, O. R. Laponina // International Journal of Open Information Technologies. – 2018. – Vol. 6, No. 7. – pp. 50-56.
5. Alghamdi, A. Designing a RESTful Northbound Interface for Incompatible Software Defined Network Controllers / A. Alghamdi, D. Paul, E. Sadgrove // SN Computer Science. – 2022. – Vol. 3. – Art. 502. – DOI 10.1007/s42979-022-01423-3.
6. Sahoo, K. S. Software Defined Network: The Next Generation Internet Technology / K. S. Sahoo, S. K. Mishra, S. Sahoo, B. Sahoo // International Journal of Wireless and Microwave Technologies. – 2017. – Vol. 7, No. 2. – pp. 13–24. – DOI 10.5815/ijwmt.2017.02.02.

7. Zaw Lin Oo. IoT Based Home Automation System using a REST API Architecture / Zaw Lin Oo, Theint Win Lai, Aung Moe // European Journal of Technique. – 2022. – Vol. 12, No. 2. – pp. 123–128. – DOI 10.36222/ejt.1018131.
8. Maurya, R. Application of Restful APIs in IOT: A Review / R. Maurya, K. A. Nambiar, P. Babbe, J. P. Kalokhe, Y. S. Ingle, N. F. Shaikh // International Journal for Research in Applied Science & Engineering Technology. – 2021. – Vol. 9, Issue II. – pp. 145–151. – DOI 10.22214/ijraset.2021.33013.
9. Pham, C. Management Architecture for Heterogeneous IoT Devices in Home Network / C. Pham, Y. Lim, Y. Tan // 2016 IEEE 5th Global Conference on Consumer Electronics (GCCE). – 2016. – DOI 10.1109/GCCE.2016.7800448.
10. Buzhin, I. G. Comparative Analysis of the REST and gRPC Used in the Monitoring System of Communication Network Virtualized Infrastructure / I. G. Buzhin, A. Yu. Derevyankin, V. M. Antonova, A. P. Perevalov, Yu. B. Mironov // T-Comm: Telecommunications and Transport. – 2023. – Vol. 17, No. 4. – pp. 50-55. – DOI 10.36724/2072-8735-2023-17-4-50-55.