ОСНОВА АРХИТЕКТУРЫ «ИНТЕРНЕТА ВЕЩЕЙ»

РОБЕРТО СИАГРИ (ROBERTO SIAGRI)
ПЕРЕВОД: АЛЕКСЕЙ ПЯТНИЦКИХ

В статье рассматривается современная архитектура решений для «Интернета вещей» (Internet of things, IoT). В качестве примера приведено готовое решение от компании Eurotech, представлены основные преимущества его применения. Подробно описаны его компоненты, такие как программное обеспечение для шлюзов и облачная платформа, являющиеся продуктами с открытым программным кодом от Eclipse Foundation.

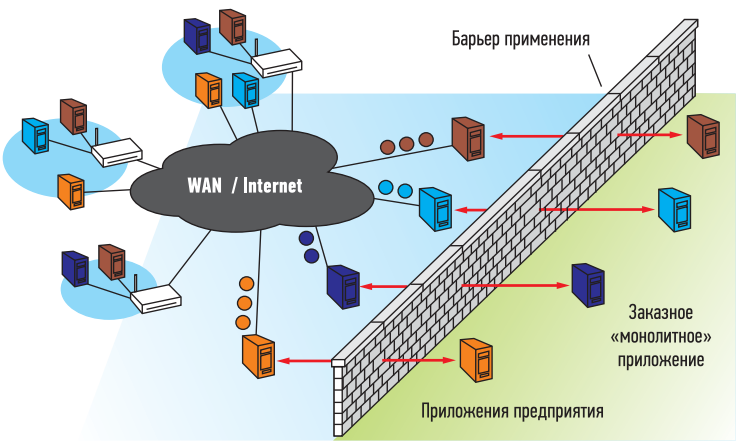

РИС. 1. ▶ Архитектура M2M позволяет сервисам (устройствам) общаться только по схеме «один с одним», что является существенным ограничением

Существующая архитектура M2M (рис. 1) позволяет интегрировать приложения в бизнес-системы предприятия по схеме «один с одним», т. е. одно приложение подключено к одной бизнес-системе. Если те же самые данные нужны второй системе, то приходится использовать вторую линию связи — и т. д. Такая архитектура негибкая и менее всего приспособлена для корпоративных систем управления предприятиями. Она является барьером для интеграции. Требуется специализированная заказная разработка для доступа к данным.

В то же время корпоративные среды управления могут добавлять и удалять информационно-коммуникационные системы без внесения изменений в архитектуру. В этом случае на первый план выходит сервисная корпоративная шина для устройств.

Сервисная корпоративная шина (англ. Enterprise Service Bus, ESB) широко применяется не только для корпоративных информационно-коммуникационных сетей, но и в мировом Интернете. Это многократно проверенная на практике концепция коммуникационной шины, которая позволяет различным приложениям и устройствам предприятия связываться между собой. Как показано на рис. 2, ESB для устройств дает возможность IoT-приложениям осуществлять связь с бизнес-средами предприятия тем же способом. В этом случае ESB становится бизнес-расширением домена «Интернета вещей» (рис. 3).

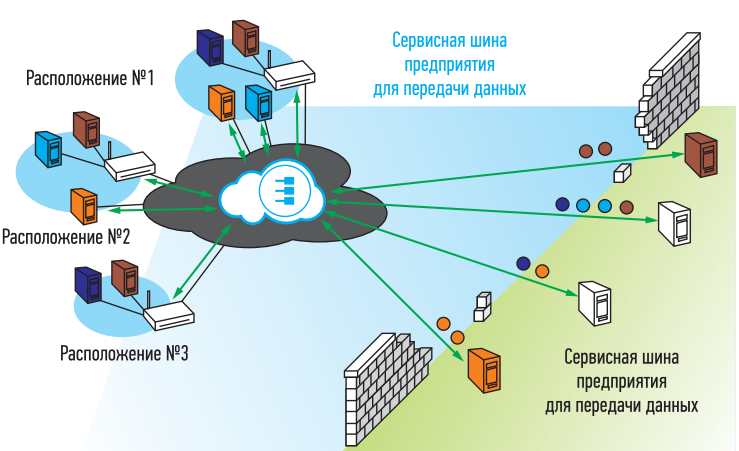

РИС. 2. ▶ Архитектура «Интернета вещей» очень гибкая: производитель и потребитель M2M-данных не привязаны друг к другу. Она поддерживает схему «многие со многими»

ESB для устройств от компании Eurotech основана на уже зарекомендовавшем себя на рынке продукте, который был разработан для осуществления связи между взаимодействующими программными приложениями в сервис-ориентированных архитектурах (англ. Service-Oriented Architecture, SOA). Такая архитектура базируется на программных компонентах, предоставляющих свои функциональные возможности в качестве сервиса другим приложениям. А если основой являются программные компоненты, то архитектура по определению очень гибкая.

ЗАКОНЧЕННОЕ РЕШЕНИЕ

Процесс работы с данными в M2M можно разбить на три основных этапа: сбор, передача и обработка данных. В «Интернете вещей» все так же, за исключением того, что информация передается через межплатформенное программное обеспечение (ПО). На рис. 4 представлено законченное решение для промышленного «Интернета вещей», где межплатформенное ПО находится в облачном решении компании Eurotech Everyware Cloud, которое является iPaaS (англ. Integration platform as a service — интеграционная платформа как сервис). Данная платформа предоставляет пользователям комбинацию облачных сер-

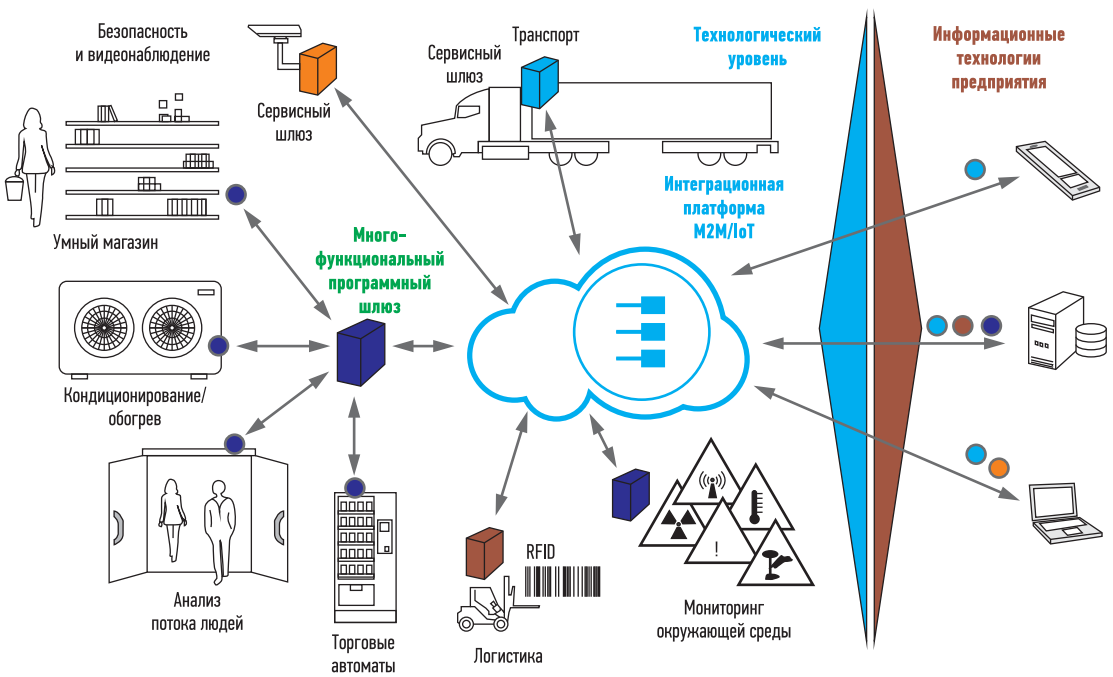

РИС. 3. ◀ Компания Eurotech построила решение ESB для устройств, позволяющее различным корпоративным приложениям работать с любыми M2M-источниками данных

висов, также называемых сервисами интеграционной платформы, для разработки и исполнения проектов, а также управления ими.

Сбор информации начинается с датчиков, которые контролируют и управляют параметрическими данными, и исполнительных устройств, передающих информацию о своей работе. Решения B2B в сегменте «Интернета вещей» обычно включают огромное количество датчиков — десятки или сотни тысяч. Eurotech разработала технологии, позволяющие быстро и легко развертывать приложения, а в дальнейшем управлять ими. Они дают возможность предоставлять экономически эффективные решения, имеющие расширенные функции масштабирования для корпоративных сетей, а также для предприятий малого и среднего бизнеса.

В зависимости от расстояния используются различные коммуникационные технологии для подключения интеллектуальных шлюзов к облачным сервисам –например, Wi-Fi для локальных систем и сотовые технологии для глобальных. Основная задача шлюзов — агрегирование данных, но они часто применяются для выполнения аналитических функций или задач предварительной обработки, в частности, для передачи данных, отвечающих заданным параметрам. Это необходимо для уменьшения объема передаваемой информации и ее нормализации, к примеру, конвертации исходных данных с датчиков в стандартный формат (рис. 5).

Комплексное решение Everyware Cloud предоставляет функциональность, требующуюся для выполнения преобразования сообщений, их маршрутизации, преобразования протоколов, нормализации данных, виртуализации сервисов, отслеживания, учета, администрирования, а также для управления жизненным циклом распределенных устройств.

Это позволяет рассматривать инфраструктуру полевого уровня, с точки зрения ИТ, как расширение для корпоративной системы управления, обеспечивающее взаимодействие между всеми компонентами с помощью коммуникационных технологий.

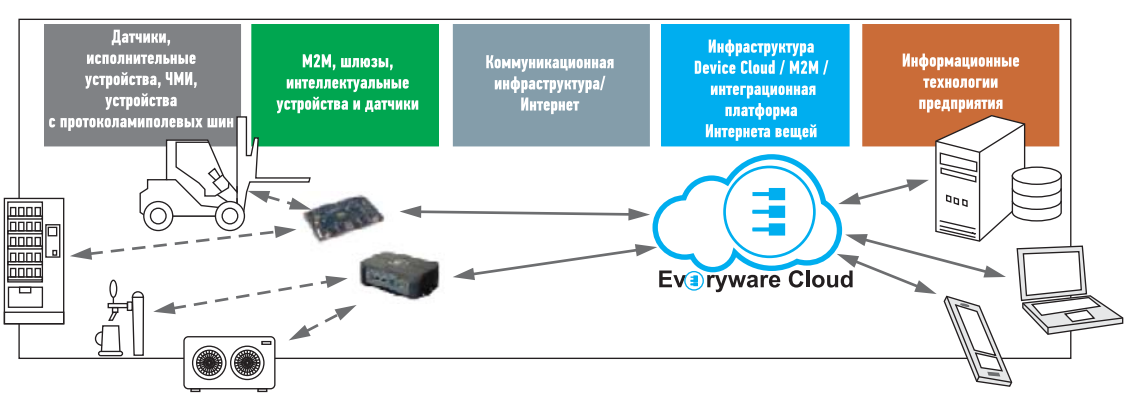

РИС. 4. ◀ Everywhere Cloud — интегрированная программная платформа, являющаяся частью Device Cloud и сервисной шиной предприятия для устройств

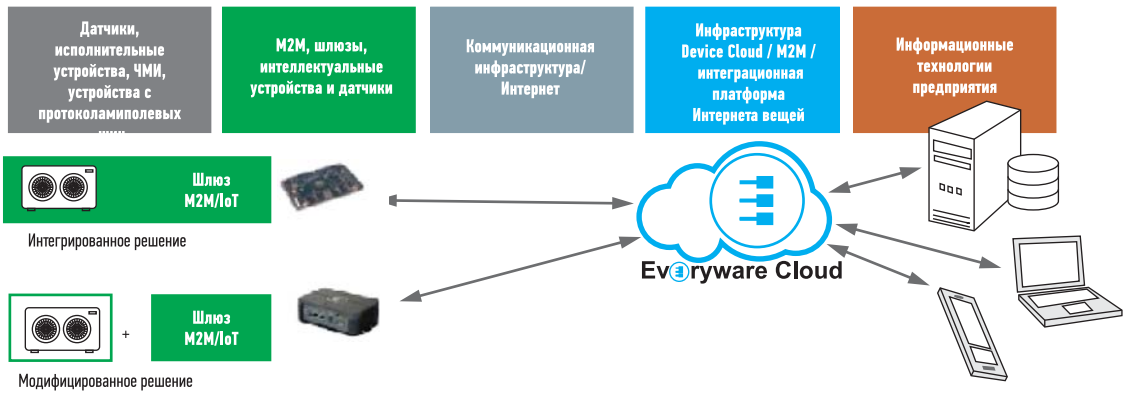

РИС. 5. ▶ Архитектура Eurotech может использоваться как для стандартных M2M-решений, так и для задач по модернизации существующих систем

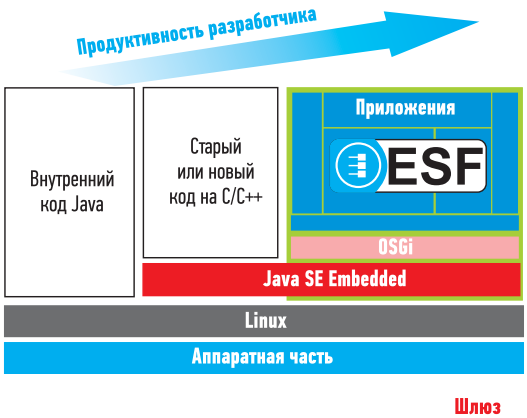

РИС. 6. ▲ ESF — межплатформенное программное обеспечение на базе Java

ИНТЕЛЛЕКТУАЛЬНЫЕ IOT-ШЛЮЗЫ С ПО JAVA-OSGI

По мнению исследовательского агентства Harbor Research, «с подключением транспортных средств к Интернету значительно увеличился поток информации, и производители были вынуждены ограничивать данные, которые необходимо отправлять в облачные сервисы». Их обработка непосредственно на транспортном средстве позволяет идентифицировать и передавать наиболее важные данные: например, с помощью установленных правил в облако отправляется информация только тогда, когда возникает механическая неисправность или обнаружены аномалии в процессе движения. Таким образом, с учетом растущих требований по компьютерным мощностям на местах, а также того, что современные программные среды дают возможность запускать множество приложений, можно отметить, что возникает ряд интересных факторов, помогающих лучше понять современную ситуацию:

- 90% всех созданных данных никогда не анализировалось;
- данные создаются в 2 раза быстрее, чем растут пропускные возможности;
- 60% данных теряют свою ценность в течение миллисекунд;
- в 2017 г. вычислительные мощности смартфонов превысят мощности серверов и систем хранения в дата-центрах.

Описанные факторы можно учитывать не только для транспортного рынка. Все эти наблюдения справедливы и по отношению ко многим другим промышленным сценариям или в тех случаях, когда требуется обработка информации на местах или аналитика в шлюзах «Интернета вещей».

Чтобы полностью соответствовать требованиям по вычислительным возможностям и в то же время контролировать и управлять работой шлюза (изменять параметры в реальном времени, обновлять ПО, производить мониторинг устройства, диагностику, обеспечивать безопасность и т. д.), было разработано ПО Java/OSGi Framework для IoT-шлюзов (рис. 6).

Версия с открытым исходным кодом под названием Kura доступна в Eclipse Foundation. ESF/Kura помогает разработчикам сфокусироваться только на приложении или аналитике, не тратить время на ключевые функции шлюза. Это высокоинтегрированное ПО имеет модульную структуру, состоящую из «строительных блоков» (рис. 7).

Такой подход дает заказчикам и разработчикам следующие преимущества:

- снижение времени на разработку → быстрая реализация проекта;
- фокус на приложении → дифференциация продуктов и предложений;
- компактный и защищенный код → высокое качество ПО;
- низкие требования к ресурсам → снижение стоимости разработки;
- аппаратная виртуализация → защита вложений;

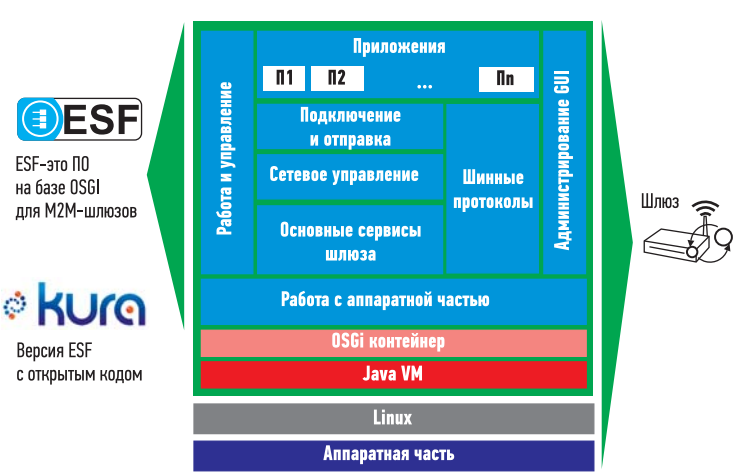

РИС. 7. ▶ Обзор функциональности Everyware Software Framework (ESF)

- детерминированное исполнение проекта → представление продукта на рынке в срок;
- базирование на стандартных продуктах → перспективные разработки, защита инвестиций;
- удаленное управление приложениями → увеличенный жизненный цикл продукта.

ESF — это промышленная версия Eclipse Kura с дополнительными возможностями по безопасности, диагностике, конфигурированию и удаленному доступу, полностью интегрируемая в платформу «Интернета вещей» Everyware Cloud (которая скоро будет доступна с открытыми кодами в Eclipse Foundation под именем Kapua).

ПОДКЛЮЧЕНИЕ IOT-ШЛЮЗА К ОБЛАЧНОМУ СЕРВИСУ

Решения на базе «Интернета вещей» создают интеграционный мост между технологическим уровнем и ИТ предприятия. При этом необходима общая платформа, способная связать все датчики и исполнительные устройства с ИТ-инфраструктурой.

Рассмотрим аналог шлюза в облаке — платформу «Интернета вещей» как сервис (iPaaS) и открытую платформу под названием Kapua.

Eclipse Kapua — это модульная iPaaS-платформа, объединяющая технологический уровень и ИТ. Она предоставляет полное управление полевыми устройствами и IoT-шлюзами, в том числе подключение, конфигурацию и управление жизненным циклом. Собираемый в реальном времени поток данных от конечных устройств можно заархивировать для дальнейшего анализа или передать в приложения верхнего уровня ИТ предприятия. Кроме того, Eclipse Kapua предусматривает веб-консоль администратора для настройки и управления, а также доступ к данным с помощью команд REST API, обеспечивая таким образом легкую интеграцию приложений.

Цель проекта Eclipse Kapua — предоставить интеграционную платформу «Интернета вещей», соответствующую следующим требованиям:

1. Платформа должна позволять подключать устройства и шлюзы «Интернета вещей» по различным протоколам. На начальной стадии добавлены протоколы, используемые в «Интернете вещей», такие как MQTT. Поддержка остальных протоколов будет реализована позднее. Уровень подключения также отвечает за аутентификацию и авторизацию устройств.
2. Платформа должна управлять полевыми устройствами. Менеджер управления устройствами (Device Manager) должен иметь возможность конфигурировать устройства, обновлять ПО и управлять устройством удаленно.
3. Устройства «Интернета вещей» должны собирать большой объем телеметрических данных.
4. Платформа «Интернета вещей» должна опираться на прочный фундамент. В частности, необходимо обеспечить управление многопользовательскими учетными записями, пользователями, разрешениями и ролями.
5. IoT-платформа должна полностью программироваться с помощью веб-сервисов REST (англ. Representational State Transfer — передача состояния представления). Желательна веб-консоль администрирования для оператора устройства.
6. Платформа «Интернета вещей» должна разворачиваться либо в облаке, либо локально.
7. Установочный пакет должен обеспечивать различные гибкие возможности развертывания.

На рис. 8 показана функциональная архитектура проекта Eclipse Kapua.

УПРАВЛЕНИЕ ДАННЫМИ

Можно идентифицировать два типа потоков данных, которые идут от шлюзов к платформе «Интернета вещей» и обратно. Первый тип — это информация, поступающая от исполнительных устройств, датчиков и т. п.; другой поток генерируется функциями управления (такими как идентификация устройств, обновление ПО, установки реального времени). Поскольку во многих архитектурах «Интернета вещей» функции управления устройствами либо пропущены, либо слабо реализованы, рассмотрим их более подробно.

УПРАВЛЕНИЕ УСТРОЙСТВАМИ

Через компоненты управления устройствами IoT-платформа может выполнять удаленное управление подключенными устройствами. Платформа предоставляет открытую сессию управления устройством, при этом не влияя на выполнение соответствующей прикладной программы. В начальной версии сессия управления устройствами базируется на открытом протоколе поверх MQTT. Он уже реализован в проекте Eclipse Kura, и с его помощью платформа может:

- анализировать и управлять конфигурацией устройства;
- управлять сервисами устройства, в том числе запуском сервиса и остановкой операций;
- управлять приложениями, в числе которых установка, обновление и удаление;
- выполнять команды операционной системы удаленно;
- получать и устанавливать атрибуты и ресурсы устройства;
- предоставлять начальную конфигурацию устройства.

В дальнейшем Eclipse Kapua может включить в себя новые протоколы управления устройствами, например стандарт LWM2M. ⬣

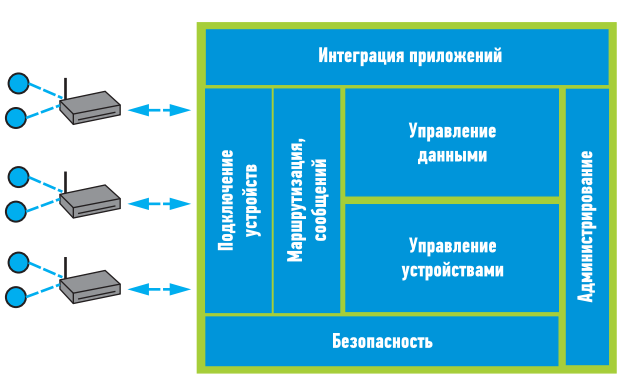

РИС. 8. ◀
Архитектура проекта Eclipse Kapua