Formal Model for Checking the Interoperability Between the Components of the IoT system

Timenko A.V.[1], Shkarupylo V.V.[2], Oliinyk A.O.[1], Hrushko S.S.[1]
[1] Zaporizhzhia National Technical University
Zaporizhzhia, Ukraine
[2] National University of Life and Environmental Sciences of Ukraine
Kyiv, Ukraine

Abstract. Today, the significant volumes of network traffic circulate through the Internet. The sources of such traffic are, in particular, the diverse territory distributed "smart" devices. The number of named devices is about billions. As a consequence, the relevance of bringing to practice the core concepts of the Internet of Things paradigm is constantly becoming more and more topical. It's bound with the problem of granting the interoperability between the components of distributed software systems, built over the aforementioned devices. The web services are typically considered as the components of the system. To this end, to establish the interoperability between the components, despite the standardization, the need for the development of effective tools and techniques, granting the interoperability between the web services, arises. The goal of the work is to increase the effectiveness of the Internet of Things system engineering process by way of checking the interoperability between the components during the designing. The goal is achieved through the development of formal model for checking the interoperability between the components of the Internet of Things system by way of model checking in an automated manner. The novelty of proposed solution is grounded on the usage of Temporal Logic of Actions, corresponding formalism and the concept of action as the basis for compact and easily reconfigurable formal specifications synthesis. The adequacy of proposed model has been proved through the case study. The verification-related time costs have been estimated.
***Keywords*:** Internet of Things, web service, formal model, specification, verification, composition, interoperability, consistency, model checking, big data.
DOI: 10.5281/zenodo.3239196

Model formal pentru verificarea compatibilității componentelor sistemului IoT

Timenko A.V.[1], Shkarupilo V.V.[2], Oleinik A.A.[1], Grushko S.C.[1]
[1]Universitatea Tehnică Națională din Zaporoje
Zaporoje, Ucraina
[2]Universitatea Națională de resurse bio și Științele Mediului din Ucraina,
Kiev, Ucraina

Rezumat. În prezent, Internetul generează sume importante de trafic, ale căror surse sunt adesea diferite tipuri de dispozitive "inteligente". Numărul de astfel de dispozitive se cifrează la miliarde. Ca urmare, implementarea paradigmei Internet of Things (IoT) devine din ce în ce mai importantă. Aceasta implică asigurarea compatibilității între componentele sistemelor software distribuite, construite pe dispozitivele care interacționează. În acest sens, rolul componentelor sistemului este adesea serviciile web. În acest sens, pentru a asigura compatibilitatea între componente, pe lângă standardizare, este necesară dezvoltarea unor instrumente și mecanisme eficiente pentru a asigura și verifica coerența interacțiunii dintre serviciile web. Scopul lucrării este creșterea eficienței procesului de dezvoltare a internetului obiectelor. Acest obiectiv este realizat prin dezvoltarea unui model formal de verificare a compatibilității componentelor sistemului Internet of Things, care este destinat să servească drept bază pentru efectuarea unei verificări automate utilizând metoda de verificare a modelului la etapa de proiectare a sistemului. Cel mai important rezultat al lucrării este acela că a fost propus un model formal al specificației sistemului IoT, care se bazează pe logica temporală a acțiunilor (Temporal Logic Actions) și pe formalismul corespunzător. Noutatea științifică a lucrării constă în utilizarea conceptului de "acțiune" al logicii temporale menționate mai sus, care a făcut posibilă obținerea unor specificații formale compacte reconfigurabile care reflectă specificul sistemului în cauză - modificarea cerințelor privind caracteristicile sistemului în funcție de schimbările condițiilor externe. Adecvarea modelului propus este confirmată de exemplul unui scenariu de domeniu. Se măsoară costurile de timp asociate verificării automate a specificațiilor, sintetizate conform modelului propus.
Cuvinte-cheie: Internet de lucruri, serviciu web, model formal, specificație, verificare, compoziție, compatibilitate, consistență, verificare model, date mari.

Формальная модель проверки совместимости компонентов IoT-системы

Тименко А.В.[1], Шкарупило В.В.[2], Олейник А.А.[1], Грушко С.С.[1]
[1] Запорожский национальный технический университет
Запорожье, Украина
[2] Национальный университет биоресурсов и природопользования Украины,
Киев, Украина

Аннотация. В настоящее время в сети Интернет генерируются значительные объемы трафика, источниками которого, зачастую, являются различного рода «умные» устройства. Количество подобных устройств исчисляется миллиардами. В результате этого все большую актуальность приобретает реализации парадигмы Интернета вещей. Это сопряжено с обеспечением совместимости между компонентами распределенных программных систем, построенных поверх взаимодействующих устройств. В роли компонентов системы при этом, зачастую, выступают веб-сервисы. В связи с этим, для обеспечения совместимости между компонентами, помимо стандартизации, возникает потребность разработки эффективных средств и механизмов обеспечения и проверки согласованности взаимодействия между веб-сервисами. Цель работы – повышение эффективности процесса разработки системы Интернета вещей. Поставленная цель достигается за счет разработки формальной модели проверки совместимости компонентов системы Интернета вещей, предназначенной служить базисом для проведения автоматизированной верификации методом проверки на модели на этапе проектирования системы. Наиболее существенный результат работы состоит в том, что была предложена формальная модель спецификации системы Интернета вещей, которая основывается на темпоральной логике действий (Temporal Logic of Actions) и соответствующем формализме. Научная новизна работы состоит в использовании концепции «действия» названной темпоральной логики, что позволило получать компактные реконфигурируемые формальные спецификации, отражающие специфику рассматриваемой системы – изменение требований к характеристикам системы в зависимости от изменения внешних условий. Адекватность предложенной модели подтверждена на примере сценария предметной области. Измерены временные затраты, сопутствующие автоматизированной верификации спецификации, синтезированной согласно предложенной модели. Эти затраты охарактеризованы как приемлемые – с учетом вычислительной сложности решаемой задачи и относительно низких вычислительных возможностей использованной аппаратной платформы. Отмечено, что названные затраты в дальнейшем могут быть снижены.

Ключевые слова: Интернет вещей, веб-сервис, формальная модель, спецификация, верификация, композиция, совместимость, согласованность, проверка на модели, большие данные.

ВВЕДЕНИЕ

В настоящее время число различных «умных» устройств (например, смартфоны, планшеты и др.) стремительно возрастает. В данном ключе, с точки зрения автоматизации, с целью интегрирования подобных устройств в рамках различного рода вычислительных процессов, потребность в освоении технологии, реализующей указанную концепцию, приобретает все большую актуальность.

В данном случае подразумевается концепция Интернета вещей (Internet of Things, IoT), которая может быть охарактеризована следующим образом: множество физических объектов, подключенных к сети Интернет, взаимодействуют, обмениваются информацией и координируют свои действия [1].

Озвученная концепция подразумевает взаимодействие миллиардов подобных объектов [2].

Существует множество различных сценариев реализации Интернета вещей: «умный дом» [3], «умный город» [4] и др. Чтоб уникальным образом идентифицировать подобные устройства в составе системы, используется технология RFID (Radio Frequency Identification) [5].

Разнообразие используемых коммуникационных протоколов и взаимодействующих устройств обуславливает возникновение проблемы интероперабельности (совместимости), т.е. способности устройств различной корпоративной принадлежности, поддерживающих, в общем случае, различные коммуникационные протоколы, взаимодействовать между собой. В данной работе такая совокупность взаимодействующих разнородных «умных» устройств рассматривается как система, синтезируемая согласно некоторому заданному (предусмотренному) сценарию использования.

Для рассуждения на тему совместимости IoT-система рассматривается с точки зрения основополагающей архитектуры. Выделяются четыре уровня архитектуры (снизу-вверх): уровень сенсоров и соединений, сетевой уровень, уровень сервисов управления и обеспечения безопасности, уровень «умных» приложений [6]. Верхний иерархический уровень можно рассматривать как предметно-зависимый – с точки зрения области применения соответствующих приложений: умный дом, умный город, умная система электроснабжения и т.д. С точки зрения масштаба и сложности (комплексности), подобные предметно-ориентированные программные решения варьируются существенным образом, однако механизм их функционирования подобен – базируется на вызовах веб-сервисов. Это означает, что реализация некоторого заданного сценария функционирования системы состоит в координировании соответствующим образом компонентов системы – веб-сервисов.

С целью обобщения, для абстрагирования от предметно-зависимых сценариев, рассмотрение названной системы в рамках работы выполняется на третьем (предпоследнем) уровне – уровне сервисов. В данном ключе IoT-система рассматривается как композиция взаимодействующих веб-сервисов [7]. Такой подход характеризуется преимуществами в следующих позициях: автоматизация, повторное использование, гибкость реконфигурирования.

Каждый отдельно взятый веб-сервис, включенный в состав композиции, рассматривается как компонент системы. Такой компонент, в свою очередь, также может являться системой, реализующей некоторую функционально завершенную часть вычислительного процесса.

Названные компоненты взаимодействуют между собой путем обмена сообщениями. В связи с этим, концепция совместимости рассматривается с позиции протоколов взаимодействия. В общем случае, проблема обеспечения совместимости может быть решена путем стандартизации [8], однако, в действительности имеет место практическое использование различных альтернативных протоколов, в том числе прикладных: MQTT (Message Queue Telemetry Transport), XMPP (Extensible Messaging and Presence Protocol), CoAP (Constrained Application Protocol) и т.д. [9]. Вместе с тем, существует множество различных подходов, ориентированных на достижение совместимости. Некоторые из них рассматриваются ниже.

Один из подходов состоит в создании и использовании механизма трансляции протоколов, предназначенного для использования в индустриальных сценариях (Industrial IoT, IIoT) [10]. Альтернативное решение заключается в использовании инструментария межплатформенного взаимодействия, где под платформой подразумевается программная система [11]. При этом оперируют понятием вертикальной совместимости. Возможное решение – система-шлюз, развернутая на мобильном устройстве и выполняющая функции интерфейса между IoT-устройством и средой взаимодействия (Интернет) [12]. Еще одно решение для обеспечения вертикальной совместимости – инструментарий HyperCat и соответствующий IoT-концентратор [13].

Для автоматизации взаимодействия типа M2M (Machine-to-Machine) предложен M2M-шлюз, предназначенный к развертыванию и использованию на мобильных устройствах [14]. В данном случае подразумевается концепция горизонтальной совместимости, когда различные приложения взаимодействуют и обмениваются данными в автоматическом режиме. Она может быть реализована путем разработки соответствующего протокола взаимодействия, охватывающего также вопросы обеспечения надлежащего уровня QoS-характеристик (Quality of Services) [15].

Компромиссный подход состоит в стандартизации наряду с использованием формальных языков описания поведений IoT-устройств. Соответствующая реализация – система EUDroid [16]. Более того, задачу проверки совместимости предлагается решать уже на этапе проектирования – оперируя моделями архитектуры IoT-системы [17].

Отдельную область исследований составляет вопрос обеспечения совместимости данных – достижения защищенного и управляемого обмена данными между программными системами [18]. Также отмечается важность проверки временной согласованности передаваемых сообщений, которая имеет место при устранении ошибок передачи [19].

Альтернативный подход – достижение совместимости на семантическом уровне –

путем обеспечения целостности семантических описаний веб-сервисов [20].

С позиции пространственной удаленности IoT-устройств, обосновывается важность проверки результирующей согласованности взаимодействия компонентов системы [21]. При этом важность поддержания временной согласованности при обмене данными между устройствами возрастает [22]. Пример соответствующей задачи – проверка согласованности применительно к «большим данным» (big data) IoT [23].

В данной работе предлагается модель, которая основывается на подходах, принятых в работах [16] и [17], а именно – на решении задачи проверки совместимости компонентов системы на этапе проектирования, с использованием формальных методов.

Цель работы – повышение эффективности процесса разработки IoT-систем. Поставленная цель достигается за счет разработки формальной модели проверки совместимости компонентов системы, предназначенной служить базисом для проведения автоматизированной верификации методом проверки на модели на этапе проектирования системы.

Научная новизна работы состоит в использовании концепции «действия» темпоральной логики действий (TLA, Temporal Logic of Actions) [24], что позволило получать компактные реконфигурируемые формальные спецификации, отражающие специфику рассматриваемой системы – изменение требований к характеристикам системы в зависимости от изменения внешних условий.

МЕТОДЫ, РЕЗУЛЬТАТЫ И ОБСУЖДЕНИЕ

В работе концепции совместимости и согласованности рассматриваются совместно. Вводится допущение, что компоненты системы совместимы, если они оперируют одинаковыми протоколами, а их взаимодействия согласованы по времени.

Рассматривая взаимодействие между некоторыми веб-сервисами путем обмена сообщениями, подразумевается горизонтальная совместимость.

Согласованность взаимодействий компонентов системы проверяется в автоматизированном режиме – с использованием формального метода проверки на модели (Model Checking).

При описании формальной модели IoT-системы используются концепции «действия» и «динамики».

Под «действием» подразумевается вызов веб-сервиса – компонента IoT-системы, под «динамикой» – последовательность таких вызовов, уникальным образом идентифицирующих некоторый вычислительный процесс – сценарий функционирования системы.

Чтобы формализовать озвученные концепции, используется темпоральная логика TLA и соответствующий формализм TLA+ [24]. Данный формализм широко используется в различных предметных областях, например, при проверке проектных решений для веб-сервисов Amazon (Amazon Web Services, AWS) [25], при разработке надежных и отказоустойчивых модулей программной платформы системы управления железнодорожным движением уровня SIL 4 (Safety Integrity Level) [26].

Математическая строгость формализма TLA+ позволяет создавать компактные и реконфигурируемые формальные описания (модели) системы.

I. МЕТОДЫ

Для формализации «действия» воспользуемся структурой Крипке [27]. Структурой Крипке M на множестве атомарных высказываний AP является структура следующего вида:

$$M = \langle S, \{s_0\}, R, L \rangle, \quad (1)$$

где S – конечное множество состояний, $s_0 \in S$ – начальное состояние, $R \subseteq S^2$ – множество переходов между состояниями, $L : S \to 2^{AP}$ – функция разметки состояний.

Пусть $s \in S$ – некоторое текущее состояние, а $s' = R(s) \in S$ – последующее состояние как результат перехода $(s, s') \in R$.

«Действие» в данном ключе – это булева функция перехода, принимающая истинное значение при переходе $(s, s') \in R$.

Для формирования множества атомарных высказываний AP введем следующие два множества:

$$V = \{v_i\}_{i=1}^{m \in N}, \quad (2)$$

$$D = \{d_1, d_2\}, \quad (3)$$

где V – множество переменных состояний, D – множество значений переменных состояний: $d_1 = 0$, $d_2 = 1$.

Каждая $v_i \in V$ представляет некоторый веб-сервис в составе композиции, который либо подлежит вызову, либо уже был вызван – в зависимости от значения из множества D.

Множество AP представим декартовым произведением:

$$AP = V \times D. \quad (4)$$

Элементы множества AP следует интерпретировать следующим образом: $(v_i,0) \in AP$ – i-й веб-сервис еще не был вызван; $(v_i,1) \in AP$ – i-й веб-сервис уже был вызван. Для акцентирования внимания на действиях, представим множество AP в виде объединения:

$$AP = AP' \cup AP'', \quad (5)$$

где $AP' = \{ap'_i\}$ – множество предусловий для действий: $ap'_i = (v_i,0) \in AP' \subset AP$ – предусловие для i-го действия; $AP'' = \{ap''_i\}$ – множество постусловий: $ap''_i = (v_i,1) \in AP'' \subset AP$ – постусловие (результат) для i-го действия.

Для представления в формальной модели совместимости двух сервисов за основу берется действие, обуславливающее переход $(s,s') \in R$. В данном ключе вводится концепция «события» – e, которое представляется в формальной модели как импликация, модифицированная темпоральным оператором Next (X) [28]:

$$e_i \equiv (ap'_i \to X\ ap''_i) \equiv (\neg ap'_i \vee X\ ap''_i), \quad (6)$$

что означает, что, если условие $ap'_i \in AP'$ истинно, тогда соответствующее условие $ap''_i \in AP''$ должно быть истинным в последующий момент модельного времени.

Аналогичным образом формализуется концепция «пустого события» – ee_i (empty event), которая означает, что некоторая переменная состояний $v_i \in V$ не меняет своего значения в последующий момент модельного времени:

$$ee_i \equiv (ap'_i \to X\ ap'_i) \equiv (\neg ap'_i \vee X\ ap'_i). \quad (7)$$

Предложенную формализацию событий (6) и (7) положим в основу спецификации перехода $(s,s') \in R$. Для этой цели рассмотрим следующий сценарий.

Сервис, представленный переменной $v_i \in V$, будем рассматривать как совместимый с сервисом, представленным переменной $v_j \in V (i \neq j)$, при условии, что имеет место соответствующее событие e_i (6). Иными словами, если сервис, представленный переменной $v_i \in V$, является инициатором взаимодействия с сервисом, представленным переменной $v_j \in V$, то соответствующее отношение можно представить оператором предшествования $\prec$: $e_i \prec e_j$, где e_i рассматривается как предусловие совместимости i-го и j-го сервисов. В таком случае будем утверждать в формальной модели, что i-й сервис совместим с j-м сервисом именно в заданном порядке вызовов.

Для создания «событийного» каркаса спецификации перехода воспользуемся тернарным оператором «?:» языка программирования Си:

$$(e_i)?e_j : ee_j, \quad (8)$$

Спецификация на основе (8) призвана служить механизмом формирования цепочек вызовов: в случае, если условие e_i (6) истинно, проверке выполнения подлежит условие e_j, в противном случае – ee_j (7).

Формализации цепочек вызовов на основе (8) будем называть «динамиками»:

$$b_k = e_1, e_2, ..., e_m. \quad (9)$$

Нижний индекс элементов цепочки событий указывает на относительный порядок возникновения событий, т.е. $e_1 \prec e_2 \prec ... \prec e_m$. События, для которых выполняется отношение $e_i \prec e_j$, будем называть «совместным» событиями.

Темпоральную спецификацию на основе выражений (6) и (9) – шаблон спецификации динамики – представим конъюнкцией:

$$f(b_k) \equiv e_1 \wedge e_2 \wedge ... \wedge e_m. \quad (10)$$

Поскольку система в общем случае может функционировать согласно различным сценариям, то с каждой динамикой целесообразно отождествлять соответствующее подмножество состояний системы переходов:

$$S = \bigcup_{k=1}^{n \in N} S_k , \qquad (11)$$

где $S_k \subseteq S$ – подмножество состояний системы переходов, ассоциированное с k-й динамикой. Понятие совместимости при этом предлагается рассматривать в узком смысле и широком смысле.

Если подразумевается проверка совместимости компонентов системы в ключе отдельно взятой динамики, имеет место постановка задачи проверки совместимости в узком смысле:

$$M, s^* \models f(b_k), \qquad (12)$$

где $s^* \in S_k$ и $s^* \notin S \setminus S_k$. Выражение (12) означает, что для структуры (1) темпоральная формула (10) должна выполняться $\forall s^* \in S_k$.

Чтобы сформулировать постановку задачи проверки совместимости в широком смысле, сформируем как дизъюнкцию темпоральную формулу, охватывающую альтернативные динамики:

$$\xi \equiv f(b_1) \vee f(b_2) \vee \ldots \vee f(b_k) \vee \ldots \vee f(b_n). \qquad (13)$$

Соответствующую задачу проверки формализуем следующим образом:

$$M, s \models \xi . \qquad (14)$$

При этом стоит отметить, что, если поведение IoT-системы характеризуется лишь одной динамикой, постановка задачи в узком смысле также является и постановкой задачи в широком смысле.

На основе введенных концепций и формализаций сформируем основополагающие блоки формальной модели: спецификацию начального состояния, спецификации переходов между состояниями, спецификации динамик и результирующую темпоральную формулу, подлежащую автоматизированной проверке – верификации методом проверки на модели согласно сформулированной постановке задачи проверки – (12) или (14).

Спецификацию начального состояния зададим конъюнкцией:

$$Init \equiv (v_1 = 0) \wedge \ldots \wedge (v_i = 0) \wedge \ldots \wedge (v_m = 0). \qquad (15)$$

Переход $(s, s') \in R$ формализуем на основе выражения (8) и конструкции IF-THEN-ELSE. При этом выделим два случая – переход из начального состояния и переход из одного из последующих состояний. В первом случае в качестве предусловия выступает спецификация начального состояния (15):

$$Act_i \equiv (v_i' = IF(Init)\, THEN\ \neg v_i\ \ ELSE\ v_i), \qquad (16)$$

где v_i' представляет переменную состояния в последующий момент модельного времени.

Во втором случае предусловием является спецификация предыдущего перехода:

$$Act_j \equiv (v_j' = IF(Act_i)\, THEN\ \neg v_j\ \ ELSE\ v_j). \qquad (17)$$

Предложенный подход к специфицированию переходов между состояниями как действий представлен на рис. 1.

Предложенный подход к специфицированию динамики состоит в следующем:

- Основываясь на (10), сформировать спецификацию динамики, где в качестве конъюнктов будут фигурировать единственное выражение (16) – инициатор цепочки переходов, и $m-1$ выражение (17).
- Сформировать результирующую темпоральную формулу, подлежащую автоматизированной проверке согласно постановке задачи (12) методом проверки на модели:

$$Spec_k \equiv Init \wedge G[Act_1 \wedge \ldots \wedge Act_m], \qquad (18)$$

где G – темпоральный оператор «Globally»; в качестве конъюнктов в квадратных скобках фигурируют спецификации переходов на основе выражений (17) и (18).

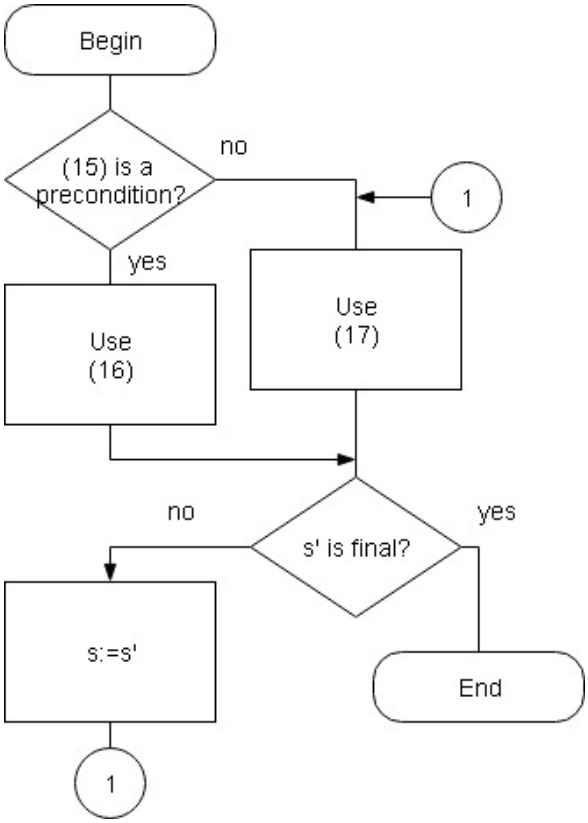


Рис. 1. Алгоритм формирования спецификаций переходов. [1]

Таким образом, постановка задачи (12) приобретает следующий вид:

$$M, s^* \models Spec_k . \quad (19)$$

В соответствии с выражением (13), в случае постановки задачи в широком смысле (14), получим результирующую темпоральную формулу следующего вида:

$$RSpec \equiv Init \wedge G\left[Spec'_1 \vee ... \vee Spec'_n\right], \quad (20)$$

где $Spec'_k \equiv Act_1 \wedge ... \wedge Act_m$ – спецификация динамики без учета спецификации начального состояния.

II. Результаты

Для проверки предложенной модели рассмотрен следующий сценарий [29]:

- Некоторый пользователь IoT-системы взаимодействует с системой посредством веб-интерфейса с целью исследования ее работы.
- Допустимы следующие варианты действий пользователя: подготовка и запуск эксперимента; получение данных (результата) проведения эксперимента и визуализация полученных данных (рис. 2).

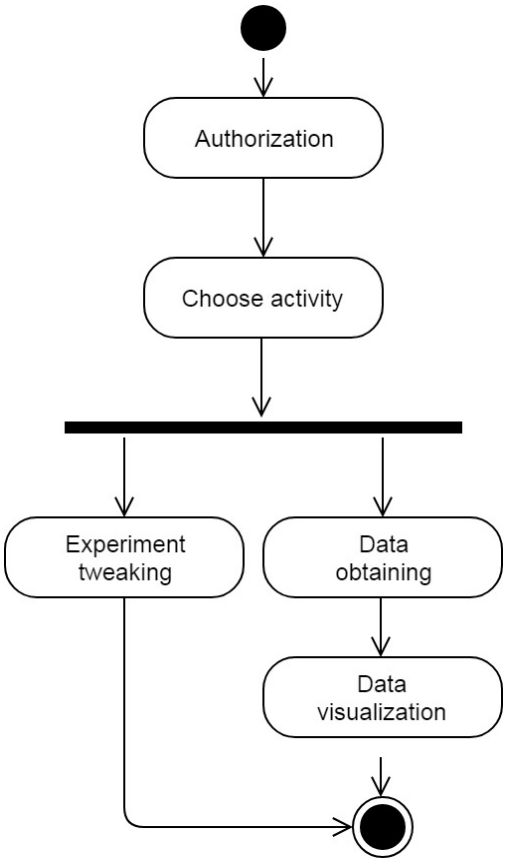


Рис. 2. Диаграмма действий для рассматриваемого сценария. [2]

На рис. 2 представлены два альтернативных сценария работы с системой, т.е. имеем две динамики.

Каждое действие реализуется соответствующим веб-сервисом. Тогда имеем следующее множество переменных состояний: $V = \{v_1, v_2, v_3, v_4, v_5\}$.

Формальная TLA+ спецификация, синтезированная согласно предложенной модели, приведена в табл. 1.

В табл. 1 в строке 6 символом [] обозначен темпоральный оператор G.

Эксперименты проведены на платформе следующей конфигурации: операционная система – MS Windows 7; процессор – Intel Celeron B815, 1.6 ГГц; оперативная память – 2 ГБ; среда исполнения Java – JRE 1.8.0_191; среда моделирования – TLA Toolbox v. 1.5.6; метод проверки на модели, реализованный в составе среды TLA Toolbox, – TLC (TLA Checker) v. 2.12. Количество достижимых состояний (в которых выполняется результирующая темпоральная формула), обнаруженных в процессе автоматизированной верификации, составило

[1,2]Appendix1

6, а сопутствующие проверке временные затраты – 2.735 с. (усредненное значение) .

Таблица 1[3].

Формальная TLA+ спецификация [4].

No.	Specification
1	VARIABLES v1, v2, v3, v4, v5 * list of variables
2	Invariant == /\ v1 \in BOOLEAN … /\ v5 \in BOOLEAN * variables invariants
3	* initial state specification as conjunction Init == v1=FALSE /\ v2=FALSE /\ v3=FALSE /\ v4=FALSE /\ v5=FALSE
4	Act_0 == v1' = IF Init THEN ~v1 ELSE v1 * specification Act_1 == v2' = IF Act_0 THEN ~v2 ELSE v2 * of Act_2 == v3' = IF Act_1 THEN ~v3 ELSE v3 * transitions Act_3 == v4' = IF Act_1 THEN ~v4 ELSE v4 * between Act_4 == v5' = IF Act_3 THEN ~v5 ELSE v5 * the states
5	Next == \/ /\ (Act_0 /\ Act_1 /\ Act_2) /\ UNCHANGED<<v4,v5>> * alternative \/ /\ (Act_0 /\ Act_1 /\ Act_3 /\ Act_4) /\ UNCHANGED<<v3>> * behaviors
6	RSpec == Init/\[][Next]_<<v1,v2,v3,v4,v5>> * resulting temporal formula

III. Обсуждение

Полученные результаты следует рассматривать как упреждающий шаг, направленный на снижение результирующих временных затрат на разработку за счет проведения формальной верификации на основе предложенной модели уже на этапе проектирования.

Полученное значение достижимых состояний системы переходов, обнаруженных в процессе автоматизированной верификации спецификации, синтезированной согласно предложенной модели, согласуется с содержимым рис. 2. Это свидетельствует в пользу адекватности предложенной модели.

Предложенная модель позволяет синтезировать компактные и реконфигурируемые спецификации IoT-системы с требуемым уровнем детализации.

Сопутствующие процессу автоматизированной верификации временные затраты можно охарактеризовать как приемлемые, учитывая относительно низкие вычислительные возможности аппаратной составляющей тестовой платформы и ресурсоемкость решения задач формальной верификации. Они могут быть дополнительно снижены, если выполнять при верификации обход состояний системы переходов не методом обхода в ширину (используется по умолчанию), а методом обхода в глубину.

ВЫВОДЫ

Таким образом, в работе предложена формальная модель проверки совместимости компонентов IoT-системы. При этом были получены следующие результаты:

- Формализована концептуальная составляющая предложенной модели – на основе структуры Крипке и выразительных возможностей формализма TLA+.
- Проверена адекватность предложенной модели – на основе сценария предметной области – вариаций сценариев взаимодействия пользователя с системой.
- Измерены временные затраты на верификацию спецификации, синтезированной согласно предложенной модели. Среднее значение составило 2.735 с. Такие затраты охарактеризованы как приемлемые.

Благодарности

Работа выполнена в рамках проекта Erasmus+ Internet of Things: Emerging Curriculum for Industry and Human Applications ALIOT Project (reference number 573818-EPP-1-2016-1-UK-EPPKA2-CBHE-JP), при участии кафедры компьютерных систем и сетей и кафедры программных средств Запорожского национального технического университета.

Работа проведена в рамках исследовательской работы «Методы и средства принятия решений для обработки данных в интеллектуальных системах распознавания образов» (номер ДБ 04927), проводимой кафедрой программных средств Запорожского национального технического университета.

[3,4]Appendix1

APPENDIX 1 (ПРИЛОЖЕНИЕ 1)

[1]**Fig. 1.** Algorithm of synthesis the specifications of transitions.

[2]**Fig. 2.** Activity diagram for the case study.

[3,4]**Table 1.** Formal TLA+ specification.

Литература (References)

[1] Al-Fuqaha A., Guizani M., Mohammadi M., Aledhari M., Ayyash M. Internet of things: a survey on enabling technologies protocols and applications. *IEEE Communications Surveys & Tutorials*, 2015, vol. 17, no. 4, pp. 2347-2376. doi: 10.1109/COMST.2015.2444095

[2] Bera S., Misra S., Vasilakos A.V. Software-Defined Networking for Internet of Things: A Survey. *IEEE Internet of Things Journal*, 2017, vol. 4, no. 6. pp. 1994-2008. doi: 10.1109/JIOT.2017.2746186

[3] Stojkoska B.L.R., Trivodaliev K.V. A review of Internet of Things for smart home: Challenges and solutions. *Journal of Cleaner Production*, 2017, vol. 140, pp. 1454-1464. doi: 10.1016/j.jclepro.2016.10.006

[4] Sotres P., Santana J.R., Sanchez L., Lanza J., Munoz L. Practical lessons from the deployment and management of a smart city Internet-of-Things infrastructure: the SmartSantander testbed case. *IEEE Access*, 2017, vol. 5, pp. 14309-14322. doi: 10.1109/ACCESS.2017.2723659

[5] Li R., Song T., Capurso N., Yu J., Couture J., Cheng X. IoT applications on Secure Smart Shopping System. *IEEE Internet of Things Journal*, 2017, vol. 4, no. 6, pp. 1945-1954. doi: 10.1109/JIOT.2017.2706698

[6] Yaqoob I., Ahmed E., Hashem I.A.T., Ahmed A.I.A., Gani A., Imran M., Guizani M. Internet of Things architecture: Recent advances taxonomy requirements and open challenges. *IEEE Wireless Communications*, 2017, vol. 24, no. 3, pp. 10-16. doi: 10.1109/MWC.2017.1600421

[7] Lemos A.L., Daniel F., Benatallah B. Web Service Composition: A Survey of Techniques and Tools. *ACM Computing Surveys (CSUR)*, 2016, vol. 48, no. 3, pp. 1-41. doi: 10.1145/2831270

[8] Mahmood K., Li X., Chaudhry S.A., Naqvi H., Kumari S., Sangaiah A.K., Rodrigues J.J.P.C. Pairing based anonymous and secure key agreement protocol for smart grid edge computing infrastructure. *Future Generation Computer Systems*, 2018, vol. 88, pp. 491-500. doi: 10.1016/j.future.2018.06.004

[9] Desai P., Sheth A., Anantharam P. Semantic Gateway as a Service Architecture for IoT Interoperability. *Proc. 2015 IEEE International Conference on Mobile Services*. New York, NY, USA, 2015, pp. 313-319. doi: 10.1109/MobServ.2015.51

[10] Derhamy H., Eliasson J., Delsing J. IoT interoperability: on-demand and low latency transparent multiprotocol translator. *IEEE Internet of Things Journal*, 2017, vol. 4, no. 5, pp. 1754-1763. doi: 10.1109/JIOT.2017.2697718

[11] Soursos S., Podnar-Zarko I., Zwickl P., Gojmerac I., Bianchi G., Carrozzo G. Towards the Cross-Domain Interoperability of IoT Platforms. *Proc. 2016 European Conference on Networks and Communication (EUCNC 2016)*. Athens, Greece, 2016, pp. 398-402. doi: 10.1109/EuCNC.2016.7561070

[12] Aloi G., Caliciuri G., Fortino G., Gravina R., Pace P., Russo W., Savaglio C. Enabling IoT interoperability through opportunistic smartphone-based mobile gateways. *Journal of Network and Computer Applications*, 2017, vol. 81, pp. 74-84. doi: 10.1016/j.jnca.2016.10.013

[13] Blackstock M., Lea R. IoT interoperability: A hub-based approach. *Proc. 2014 International Conference on Internet of Things (IOT)*. Cambridge, MA, USA, 2014, pp. 79-84. doi: 10.1109/IOT.2014.7030119

[14] Pereira C., Pinto A., Aguiar A., Rocha P., Santiago F., Sousa J. IoT interoperability for actuating applications through standardised m2m communications. *Proc. 2016 IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*. Coimbra, Portugal, 2016, pp. 1-6. doi: 10.1109/WoWMoM.2016.7523564

[15] Ferrari P., Flammini A., Sisinni E., Rinaldi S., Brandao D., Rocha M.S. Delay estimation of Industrial IoT applications based on messaging protocols. *IEEE Transactions on Instrumentation and Measurement*, 2018, vol. 67, no. 9, pp. 2188-2199. doi: 10.1109/TIM.2018.2813798

[16] Buono P., Cassano F., Legretto A., Piccinno A. EUDroid: A formal language specifying the behaviour of IoT devices. *IET Software*, 2018, vol. 12, no. 5, pp. 425-429. doi: 10.1049/iet-sen.2017.0347

[17] Sosa-Reyna C.M., Tello-Leal E., Lara-Alabazares D. Methodology for the model-driven development of service oriented IoT applications. *Journal of Systems Architecture*, 2018, vol. 90, pp. 15-22. doi: 10.1016/j.sysarc.2018.08.008

[18] Nawaratne R., Alahakoon D., De Silva D., Chhetri P., Chilamkurti N. Self-evolving intelligent algorithms for facilitating data interoperability in IoT environments. *Future Generation Computer Systems*, 2018. vol. 86, pp. 421-432. doi: 10.1016/j.future.2018.02.049

[19] Zhou Q., Simmhan Y., Prasanna V. Knowledge-infused and consistent Complex Event Processing over real-time and persistent streams. *Future*

Generation Computer Systems, 2017, vol. 76, pp. 391-406. doi: 10.1016/j.future.2016.10.030

[20] Hur K., Chun S., Jin X., Lee K.-H. Towards a semantic model for automated deployment of IoT services across platforms. *Proc. 2015 IEEE World Congress on Services*. New York, NY, USA, 2015. pp. 17-20. doi: 10.1109/SERVICES.2015.11

[21] Beal J., Viroli M., Pianini D., Damiani F. Self-adaptation to device distribution in the Internet of Things. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 2017, vol. 12, no. 3, pp. 1-29. . doi: 10.1145/3105758

[22] Shi X., Fan L., Ling Y., He J., Xiong D. Dynamic and quantitative method of analyzing clock inconsistency factors among distributed nodes. *Arabian Journal for Science and Engineering*, 2015, vol. 40, no. 2, pp. 519-530. doi: 10.1007/s13369-014-1545-1

[23] Liu F., Yang Y. A two-tier replication framework and its consensus protocol for big data storage of internet of things. *Journal of Internet Technology*, 2016, vol. 17, no. 7, pp. 1453-1460.

[24] Lamport L. *Specifying systems: The TLA+ language and tools for hardware and software engineers*. Boston, Addison-Wesley, 2002. 382 p.

[25] Newcombe C., Rath T., Zhang F., Munteanu B., Brooker M., Deardeuff M. How Amazon web services uses formal methods. *Communications of the ACM*, 2015, vol. 58, no. 4, pp. 66-73. doi: 10.1145/2699417

[26] Resch S., Paulitsch M. Using TLA+ in the development of a safety-critical fault-tolerant middleware. *Proc. of 2017 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, Toulouse, France, 2017, pp. 1-7. doi: 10.1109/ISSREW.2017.43

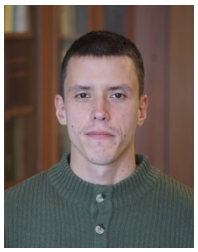
[27] Clarke E.M., Grumberg O., Peled D.A. *Model checking*. Massachusetts, MIT Press, 2001. 314 p.

[28] Shkarupylo V.V., Tomicic I., Kasian K.M., Alsayaydeh J.A.J. An Approach to increase the Effectiveness of TLC Verification with Respect to the Concurrent Structure of TLA+ Specification. *International Journal of Software Engineering and Computer Systems*, 2018, vol. 4, no. 1, pp. 48–60. doi: 10.15282/ijsecs.4.1.2018.4.0037

[29] Sanchez L., Munoz L., Galache J. A., Sotres P., Santana J. R., Gutierrez V., Ramdhany R., Gluhak A., Krco S., Theodoridis E., Pfisterer D. SmartSantander: IoT experimentation over a smart city testbed. *Computer Networks*, 2014, vol. 61, pp. 217-238. doi: 10.1016/j.bjp.2013.12.020

Сведения об авторах.

Тименко Артур Валентинович. Ассистент кафедры компьютерных систем и сетей Запорожского национального технического университета. Область научных интересов – Интернет вещей; компьютерные сети.
E-mail: timenko.artur@gmail.com



Шкарупило Вадим Викторович. К.т.н., доцент, доцент кафедры компьютерных систем и сетей Национального университета биоресурсов и природопользования Украины. Область научных интересов – Интернет вещей; формальные методы.
E-mail: shkarupylo.vadym@gmail.com

Олейник Андрей Александрович. К.т.н., доцент, доцент кафедры программных средств Запорожского национального технического университета. Область научных интересов – искусственный интеллект, нейронные сети, анализ данных. E-mail: olejnikaa@gmail.com

Грушко Светлана Сергеевна. К.т.н., старший преподаватель кафедры компьютерных систем и сетей Запорожского национального технического университета. Область научных интересов – проектирование устройств управления на ПЛИС; интерфейсы компьютерных систем.
E-mail: grushko_ss@i.ua