

# МОДЕЛЬ ВЫБОРА ШАБЛОНА ПРОГРАММНОЙ АРХИТЕКТУРЫ И ТАКТИК ПРОЕКТИРОВАНИЯ ДЛЯ СИСТЕМ ИНТЕРНЕТА ВЕЩЕЙ

Ю. В. Ядгарова\*, В. В. Таратухин\*\*

\*Московский государственный технический университет им. Н. Э. Баумана

\*\*Национальный исследовательский университет «Высшая школа экономики»

Поступила в редакцию 21.06.2019 г.

**Аннотация.** В статье приведена аналитическая модель и метод выбора базового шаблона программных архитектур и тактик проектирования для систем Интернета Вещей. Обобщено понятие IoT-технологий, показана вертикальная и горизонтальная классификация подобных систем, выделены основные значимые параметры качества и приведены методы по их достижению. Необходимые параметры качества программных систем достигаются реализацией базового шаблона программной архитектуры и сопутствующих тактик проектирования. В работе представлена аналитическая модель зависимости трудоемкости проекта, рассчитанной по методике СОСОМО II от используемых элементов программной архитектуры. Данная модель позволяет путем минимизации функции трудоемкости выбирать наиболее подходящие для конкретного типа проекта шаблоны архитектуры и тактики на ранних этапах проектирования. Указанный подход позволяет минимизировать ошибки в построении программной архитектуры на начальном этапе проекта при выборе шаблона IoT-архитектуры, когда в наличии имеются только требования и предпосылки к созданию системы. В качестве результата приведена оценка шаблона архитектуры с помощью данного подхода систем превентивного обслуживания. Подход может быть использован для достижения требуемых параметров качества системы и снижения стоимости проекта, а также при разработке работоспособных прототипов в сжатые сроки.

**Ключевые слова:** Интернет вещей, качество систем Интернета Вещей, архитектура Интернета Вещей, проектирование архитектуры ПО, параметры качества архитектуры ПО, тактики проектирования, шаблоны архитектуры.

## ВВЕДЕНИЕ

Термин «Интернет Вещей» был впервые упомянут в 1999 Кевином Эштоном, основателем MIT Auto-ID центра [1]. В последние годы технологии Интернета Вещей набирают популярность во многих сферах науки и техники. Однако наряду с этим, зачастую определение до сих пор не несет ясного смысла для пользователей данной технологии.

На международном уровне данная концепция приобретает сформировавшиеся черты, что можно проследить по появлению четкой классификации типов подобных систем, стандартов взаимодействия устройств, а также выделению фундаментальных характеристик технологии в соответствии с рекоменда-

циями Международного Союза Электросвязи (МСЭ-Т). Однако в основе построения большинства типов систем IoT (Internet of Things), так или иначе лежат принципы системного анализа.

В России на уровне федеральных органов власти проблематика передовых производственных технологий (особенно технологий Промышленного Интернета, отнесенного в данной работе к одному из типов систем Интернета Вещей) стала активно обсуждаться и рассматриваться в качестве приоритетного направления развития промышленности с 2013 г. Одним из показателей, характеризующим потенциал развития технологии, является ее общая выгода для экономики. Другим важным показателем важности развития является число подключенных к системам Интернета Вещей устройств, по разным

прогнозам, достигающих 50 млрд устройств по всему миру к 2025 году.

С точки зрения конкурентоспособности предприятий, внедряющих подобную концепцию, количество подобных сетей имеет тенденцию расти ввиду того, что положительное влияние систем Интернета Вещей на различные показатели эффективности было неоднократно доказано успешными проектами (можно рассмотреть системы «Умный дом», и, к примеру, значительное повышение уровня безопасности подобного жилища). В то же время, принимая во внимание то, что системы Интернета Вещей можно с уверенностью отнести к сложным и то, что требования к достигаемым параметрам таких комплексов сильно зависят от отрасли, достижение требуемых атрибутов с помощью только сетевых алгоритмов взаимодействия не представляется возможным. Особенно остро проблема выбора программной архитектуры, так как в конечном итоге, ошибка на управляющем уровне сводит на нет работу коммуникационных и сетевых алгоритмов. Даже на примере двух типов систем Интернета Вещей: «Умный город» (миллионы устройств, широкая география распространения) и «Умная медицина» (небольшое количество устройств, ограниченная локация), четко прослеживаются различные цели и, как следствие, различные требуемые достигаемые системные параметры при проектировании системы. Ошибки в построении программной архитектуры, как правило, приводят к неработоспособности или экономической необоснованности внедрения системы, ухудшению эффективности работы, потере времени и рабочих ресурсов и в конечном счете могут отрицательно повлиять на конкурентоспособность предприятия на рынке.

В работе предлагается подход позволяющий минимизировать ошибки в построении программной архитектуры на начальном этапе проекта при выборе шаблона IoT-архитектуры, когда в наличии имеются только требования и предпосылки к созданию системы.

Среди определений, относящихся к Интернету Вещей, попадают термины «глобальный», «вездесущий». Это логично вы-

текает из того, что Интернет также сейчас стал широко распространен. Среди терминов встречается упоминание «всеобъемлющих» или «всепроницающих» систем, обладающих следующими характеристиками [2]:

- **Неявное построение систем.** Подобные системы сложно выделить в одну, они состоят из множества интернирующихся модулей.

- **Сетевая модель взаимодействия.** Устройства соединены цельной коммуникационной инфраструктурой.

- **Связь многих со многими.** Устройства и пользователи, а также устройства между собой могут быть связаны отношением «много-ко-многим».

- **Постоянная доступность.** Для устройств необходимо постоянное включение в сеть чтобы принимать команды в любой момент времени.

- **Распределенность.** Комбинированный эффект от распределенных данных и устройств.

- **Чувствительность к окружающей среде.** Устройство осведомлено об окружающей среде.

- **Адаптивность.** Активность системы зависит скорее от внутренних системных решений, чем от вмешательства пользователя.

- **Обработка естественного языка.** С точки зрения взаимодействия с человеком, система должна быть способна обрабатывать естественные сигналы.

При написании данной статьи, будем придерживаться определения Международного Союза Электросвязи, и считать Интернет Вещей *глобальной инфраструктурой для информационного общества, которая обеспечивает возможность предоставления более сложных услуг путем соединения друг с другом (физических и виртуальных) вещей на основе существующих и развивающихся функционально совместимых информационно-коммуникационных технологий.* [3].

## 1. АНАЛИЗ РЕЗУЛЬТАТОВ ПРЕДШЕСТВУЮЩИХ РАБОТ

Наибольшую популярность системы Интернета Вещей приобрели в 2010–2017 годах,

с удешевлением конечных устройств, появлением технологий для безопасной и быстрой передачи данных, а также с развитием интернета и протоколов взаимодействия. Данные события благотворно сказались на появлении большого числа относительно недорогих и простых конечных устройств, которые в короткие сроки можно было собрать в готовое решение. Именно этим объясняется повсеместный рост аппаратных и программных систем для поддержки подобных решений (печатные платы Arduino Starter Kit, Raspberry Pi, системы «Умный дом»). Однако, наряду с простыми решениями, на рынке также стали появляться профессиональные системы с достаточно сложной и разветвленной логикой, к появлению которых причастны уже целые компании и международные комитеты (к примеру – концепция Промышленности 4.0 и Industrial Internet Consortium, Консорциум Промышленного Интернета). Подобные системы имеют сложную программную архитектуру и логику работы, что позволяет им одновременно получать данные и управлять многими сложными устройствами, сохраняя при этом возможность расширения и взаимодействия в реальном времени. Наряду с этим, в настоящий момент создано множество классификаций систем Интернета Вещей, отражающие функциональные и структурные характеристики систем. В приведенном исследовании частично применяется отсылка к промышленной классификации, однако затем на ее базе создана своя классификация систем Интернета Вещей.

Одной из популярных классификаций систем IoT является классификация Postscapes [4], которая делит все множество решений на шесть различных категорий («Умный дом», «Умный транспорт», «Умное здравоохранение», «Умные продажи», «Умный транспорт», «Умный дом» и промышленные приложения). Данная классификация практически полностью повторяет классификацию, разработанную в ходе проекта IoT-A [5], за исключением того, что IoT-A также включает Умную Энергетику и Умное здравоохранение. Некоторые из категорий перекрывают друг друга, поэтому в качестве одной из основных приведем классификацию, предложенную одним из крупнейших производителей ПО. В данной классификации системы разделены на «горизонтальные» и «вертикальные» решения на основе включения уровней модели OSI [6] (рис. 1).

К вертикальным решениям относят отраслевые решения, призванные внести функциональность Интернета Вещей в определенные отрасли, и охватывающие все уровни модели OSI, начиная от аппаратного уровня, и заканчивая аналитическими модулями. Их особенностью является фокус на конкретную область (к примеру, в случае финансового решения данные модули могут содержать библиотеки для анализа и статистической обработки финансовых данных). Разработкой вертикальных решений в основном занимаются компании, специализирующиеся на производстве решений для данных областей (к примеру, SAP MII – SAP Manufacturing

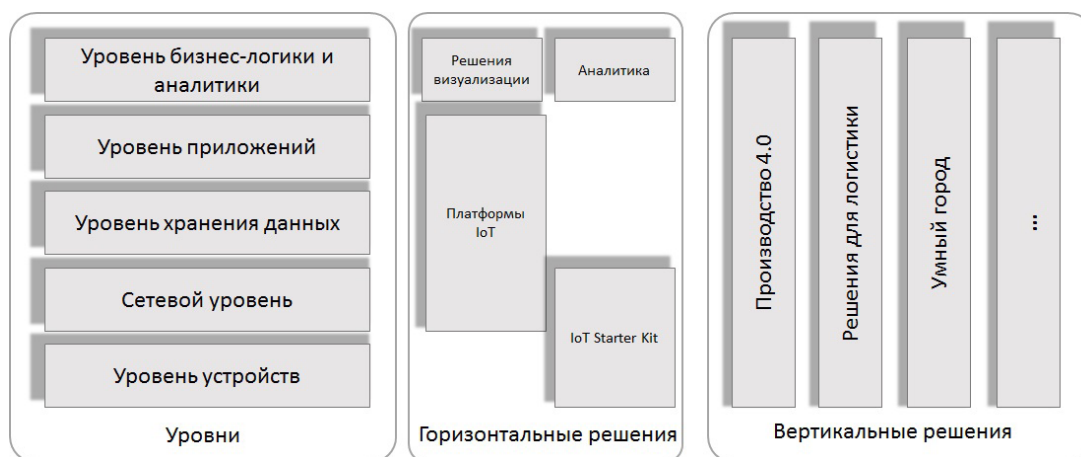


Рис. 1. Вертикальные и горизонтальные решения IoT

Integration and Intelligence) или решения «Умного Транспорта», предлагаемые Google.

К горизонтальным решениям относятся системы, не привязанные к какой-либо определенной отрасли, но помогающие построить взаимодействие аппаратно-программного обеспечения. К примеру, на сетевом уровне и уровне физических устройств модели OSI поставляются системы IoT Starter Kit, призванные облегчить построение физического уровня системы Интернета Вещей. Так называемые «Платформы промышленного интернета» в свою очередь, работают начиная с сетевого уровня и затрагивают три уровня модели OSI [5].

Внутри программной части систем Интернета Вещей существуют различия в построении, исходящие из их функциональных и структурных требований. К примеру, среди «Вертикальных» продуктов существуют решения, предназначенные для логистики, предиктивного обслуживания, промышленности, медицины и т. д. Очевидно, что, к примеру, требуемый параметр масштабируемости у данных типов систем будет иметь различное значение для систем Умной медицины (где количество устройств условно ограничено измерениями на теле человека) и систем превентивного обслуживания, где множество устройств могут быть разбросаны по всему миру, и количество этих устройств непрерывно растет.

Некоторые исследователи [7, 8] сходятся во мнении, что для приложений Интернета Вещей не может быть единого стандарта программной архитектуры, так как сам по себе термин не несет сколько-нибудь контекстной информации для принятия даже предварительных проектировочных решений. Однако в данной статье предпринята попытка опровергнуть данное мнение путем классификации различных типов IoT-решений и создания средства поддержки проектирования для конкретных типов систем Интернета Вещей. Подобная методика применительно к системам промышленного Интернета Вещей, была разработана Консорциумом Индустриального Интернета. Среди опубликованных документов можно выделить Industrial Internet

Reference Architecture [9] – стандарт, регламентирующий уровни построения, сложность, технические детали и особенности внедрения подобных систем на производстве. Указанный документ принимали к сведению большинство западных компаний при разработке систем промышленного Интернета Вещей, что показало высокую применимость методологии на практике.

## 2. МАТЕРИАЛЫ И МЕТОДЫ ИССЛЕДОВАНИЯ

В настоящем разделе рассмотрена постановка задачи проектирования программной архитектуры в соответствии с требованиями, а также модель выбора базового шаблона и тактик проектирования, удовлетворяющих заданным параметрам.

### 2.1. Стандарты и характеристики качества систем интернета вещей

В стандарте ГОСТ ISO/IEC 25010 2015 [10] для программных систем выделено восемь атрибутов качества, которые в свою очередь, также состоят из нескольких атрибутов (рис. 2).

Исследования, проводимые в Институте Архитектуры систем Интернета Вещей, также выделяют несколько критериев качества, реализация которых критически важна для подобных платформ [5]. В их число входят интероперабельность, способность эволюционировать производительность, масштабируемость, готовность, устойчивость, информационная безопасность, конфиденциальность.

Так как в данной работе качество системы оценивается с точки зрения стандарта, предложено соответствие части вышеуказанных параметров одному или нескольким атрибутам качества в терминах ГОСТ ISO/IEC 25010 2015.

Способность эволюционировать по своей природе можно определить как модифицируемость системы, в то время как масштабируемость является следствием характеристики, определяемой Стандартом как «Потенциальные возможности». Свойство является одним из характеристик-составляющих Произво-





Рис. 2. Модель качества продукта

дительности. Устойчивость программной системы определим через комбинацию Отказоустойчивости и Восстанавливаемости (характеристики-составляющие Надежности). Таким образом, можно в целом заменить оценку Устойчивости и Готовности оценкой Надежности из Стандарта. Конфиденциальность в данном случае может быть рассмотрена как под-характеристика Информационной безопасности.

Таким образом, в итоговое множество рассматриваемых атрибутов качества включены:

- Интероперабельность;
- Модифицируемость;
- Производительность;
- Масштабируемость;
- Надежность;
- Информационная безопасность.

## 2.2. Особенности проектирования программных архитектур

Говоря о структурных элементах типов программных архитектур, выделяют понятие *шаблонов архитектуры*, представляющих собой компоненты и связи между ними (элементы в терминах типа архитектуры). Другими словами, шаблон архитектуры представляют собой абстракцию структуры программной системы, состоящую из компонентов и связей между ними.

Приведем некоторые наиболее полно представленные в литературе основные архитектурные шаблоны:

- Клиент-серверная архитектура;
- Архитектура «Точка-точка»;

- Архитектура «Каналы и фильтры»;
- Событийно-ориентированная архитектура;
- Архитектура «Издатель-подписчик»;
- Сервисно-ориентированная архитектура (SOA);
- REST-архитектура;
- Архитектура слоев;
- Облачная архитектура.

Обращаясь к реализации существующих архитектур, можно утверждать, что каждый шаблон архитектуры может по построению удовлетворять некоторым параметрам качества (примером может служить хорошая масштабируемость Клиент-серверной архитектуры. Для остальных атрибутов введем понятие *тактики проектирования*. Тактика проектирования (типовой сценарий) – проекторное решение, воздействующее на достижение определенного атрибута качества. Реализация тактики всегда меняет ответ системы на конкретное воздействие.

На рис. 3 показана связь указанных понятий в терминах UML 2.

## 2.3. Модель выбора шаблона программной архитектуры

Задача выбора базового шаблона архитектуры и тактик для систем Интернета Вещей состоит в нахождении такой совокупности компонентов и связей между ними для базового шаблона и множества тактик, реализация которых приводит к достижению требуемых функциональных параметров и параметров качества системы при минимизации трудоемкости.

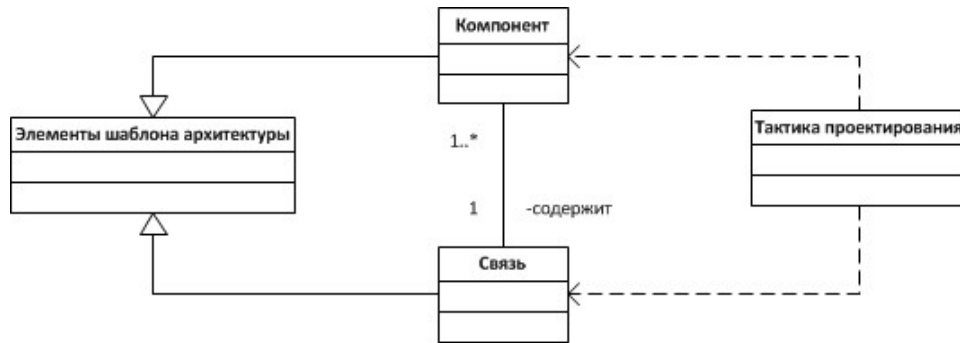


Рис. 3. Элементы шаблона архитектуры (диаграмма классов)

Полагаем, что критерием достижения параметра качества программной системы является реализация в выбранной архитектуре множества тактик проектирования  $T = t_i$ ,  $i = 1, 2, \dots, n$ , отвечающих за упомянутый параметр качества. При этом в литературе подчеркивается [11], что некоторые шаблоны программных архитектур по своему исходному построению удовлетворяют определенным параметрам качества. Данное условие выразим с помощью лингвистической переменной  $K_{ij}$ , определяющей соответствие базового шаблона архитектуры  $i$  требуемому параметру качества  $j$ :

$$K_{ij} = \begin{cases} BC | \text{тип архитектуры удовлетворяет} \\ \quad \text{по параметру качества} \\ CP | \text{тип архитектуры нейтрален} \\ \quad \text{к параметру качества} \\ NZ | \text{тип архитектуры не удовлетворяет} \\ \quad \text{по параметру качества} \end{cases}$$

В случае, если базовый шаблон архитектуры содержит высокую оценку по требуемому параметру качества, реализация тактик проектирования для данного параметра не требуется, что в свою очередь представим булевой переменной  $x_{ij}$ , значение которой равно:

$$x_{ij} = \begin{cases} 1 | \text{реализация тактик не требуется} \\ 0 | \text{реализация тактик необходима.} \end{cases}$$

Таблица соответствия базовых типов архитектур основным параметрам качества систем Интернета Вещей приведена в табл. 1.

В качестве оценки трудоемкости, среди множества аналитических методик оценки, выбрана наиболее применимая к начальной стадии проекта: СОСОМО II (англ.

COConstructive COst MOdel – Конструктивная Модель Стоимости) [12, 13].

Для решения задачи выбора шаблона архитектуры и тактик проектирования, введем понятие множества параметров качества системы, т. е совокупность параметров качества, зависящую от типа системы и требований  $S_i = S(i, t)$ , где  $i = 1..k$  – параметр качества,  $t = 1..n$  – тип системы. Каждый параметр качества характеризуется либо реализацией его в базовом шаблоне архитектуры, либо суммой тактик проектирования, необходимых для его достижения на базовом ша-

блоне:  $S_i = \begin{cases} x_{ij} \\ \sum_{k=1}^m T_k \end{cases}$ , где  $T_k$  – тактика проектирования,  $m$  – количество тактик, необходимое для реализации параметра,  $x_{ij} = 1$ , если реализация тактики проектирования для достижения не требуется,  $x_{ij} = 0$  в обратном случае.

Тактика проектирования в свою очередь может быть представлена как модификация элементов архитектурного шаблона (компонентов и связей между ними).

Однако реализация одних и тех же тактик проектирования для различных базовых шаблонов архитектуры может существенно отличаться, а следовательно, отличаться будет и трудоемкость реализации. Для того, чтобы учесть данное ограничение, введем понятие типа изменения шаблона архитектуры тактикой. Тип изменения шаблона – понятие, отражающее сущность модификации шаблона архитектуры и, следовательно, влияющие на трудоемкость. Каждая тактика может вносить от одного, до нескольких типов изменений. Подсчитаем с помощью методики

Соответствие типов архитектур параметрам качества

	Интероперабельность	Модифицируемость	Производительность	Масштабируемость	Надежность	Информационная безопасность
Клиент-сервер	1	1	0	0	0	0
Точка-точка	0	0	0	1	0	0
Каналы и фильтры	0	0	1	0	1	0
Событийно-ориентированная	0	1	0	0	0	0
Издатель-подписчик	0	1	0	0	0	0
Сервисно-ориентированная	1	1	0	0	0	0
REST	0	1	0	1	0	0
Архитектура слоев	0	0	0	0	0	0
Облачная архитектура	1	1	1	1	0	0

СОСОМО II относительную трудоемкость реализации добавления компонента в базовый шаблон архитектуры, состоящий из  $n$  компонентов.

Трудоемкость проекта задается формально как функция зависимости от количества тысяч строк кода:

$$Q = \alpha \cdot (KLOC)^b \cdot EAF, \quad (1)$$

где  $Q$  – трудозатраты, выраженные в человеко-месяцах,  $EAF$  – фактор корректировки трудозатрат в зависимости от среды,  $\alpha$  – фактор, зависящий от детальности оценки (постоянная величина, для предварительной оценки равная 2,94),  $b = B + 0,01 \cdot \sum_{j=1}^5 SF_j$ , где  $SF_j$  – факторы масштаба. Тогда исходную трудоемкость реализации можно представить как  $Q = \alpha \cdot (KLOC_0)^b \cdot EAF$ . Так как детальность оценки и среда остаются неизменными, реализация дополнительного компонента зависит от количества тысяч строк кода KLOC реализации и фактора масштаба  $b$ . Трудоемкость готового проекта, выраженная через сумму трудоемкостей компонентов:

$$Q = \sum_{i=1}^n Q_i = \sum_{i=1}^n \alpha \cdot (KLOC)_i^b \cdot EAF =$$

$$= \alpha \cdot EAF \cdot \sum_{i=1}^n KLOC_i^b.$$

Следует упомянуть, что существенным допущением при данном преобразовании является то, что при реализации каждого последующего компонента фактор масштаба остается неизменным.

Выразим также трудоемкость разработки одного модуля через трудоемкость всего проекта:  $Q_i = \alpha \cdot \left( \frac{KLOC}{n} \right)^b \cdot EAF$ , где  $n$  – количество модулей в проекте. Тогда, принимаем  $t = \frac{KLOC}{n}$ , где  $t$  – усредненное количество строк кода на один модуль. Тогда, переписывая выражение (1), получаем трудоемкость проекта как:

$$Q = \alpha \cdot EAF \cdot n \cdot t^b. \quad (2)$$

Возвращаясь к типам изменения шаблона архитектуры, в ходе исследования было выделено пять основных способов модификации. Данные способы приведены в табл. 2 и отражают характер изменений. Приведенные значения трудоемкости в свою очередь выражены как функция от величины  $t$ . Расчет, к примеру, величины Доб<sub>кн</sub> (Добавление нового совместимого компонента в шаблон), заклю-

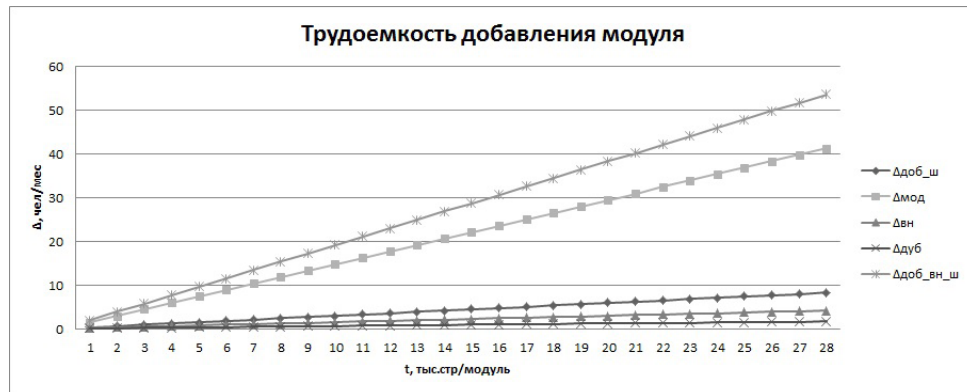


Рис. 4. Трудоемкость изменения шаблона архитектуры

Таблица 2

Типы изменения шаблона тактикой

№ п/п	Тип изменения	Трудоемкость	Описание
1	Добавление компонента в шаблон Доб <sub>кш</sub>	Низкая	Добавление нового совместимого компонента в шаблон, связь с другими. Структура остается прежней,
2	Модификация компонента Мод <sub>к</sub>	Средняя – Высокая	Меняется структура компонентов шаблона. Влияет на остальные компоненты.
3	Реализация внутри компонента Вн <sub>к</sub>	Низкая	Меняется один из компонентов шаблона
4	Дублирование компонента Дуб <sub>к</sub>	Низкая	Дублируется один из компонентов, добавляются связи.
5	Добавление компонента вне шаблона Доб <sub>квш</sub>	Высокая	Добавляется компонент, по структуре не подходящий шаблону.

чается в следующем. При наличии базового шаблона архитектуры с известными величинами  $n$  и  $t$ , добавление еще одного компонента меняет известную трудоемкость (2) на величину, равную:

$$\Delta_{доб} = \alpha \cdot EAF \cdot 1 \cdot t^b.$$

На основе расчета для каждого типа изменения построен пример график зависимости величины трудоемкости изменения от  $t$  (рис. 4). В данном примере как параметр взято значение  $n = 10$  (количество компонентов), а  $t_{мод} = 0,5t$ .

Далее, введем понятие *связанности тактик*. Две тактики будем называть *связанными*, если для их реализации можно использовать общие компоненты и связи между ними. Наличие связанной реализованной тактики

меняет тип изменения шаблона архитектуры в сторону уменьшения трудоемкости.

Таким образом, общая трудоемкость проекта для выбранного базового типа состоит из базовой относительной трудоемкости реализации шаблона (выраженной через величину  $t$ ) и сумму относительных трудоемкостей  $n$  тактик проектирования:

$$Q = Q_b(t) + \sum_{i=1}^n Q_i(t, r) = \alpha \cdot EAF \left( n \cdot t + \sum_{i=1}^n Q_i(t, r) \right), \quad (3)$$

где  $r$  – тип изменения по табл. 2. Величина  $t$  является зависящей как от шаблона архитектуры, так и от реализации компонента, и рассчитывается с помощью метода функцио-



Оценка шаблона архитектуры «Каналы и фильтры»

№ п/п	Тактика	Базовое	Доб <sub>кш</sub>	Мод <sub>к</sub>	Вн <sub>к</sub>	Дуб <sub>к</sub>	Доб <sub>квш</sub>
1	Обнаружение дефектов (Н)			14,7t			
2	Восстановление после сбоев (Н)			14,7t		0,588t	
3	Предотвращение ошибок (Н)		2,94t				
4	Контроль потребления ресурсов (П)	1					
5	Управление ресурсами (П)	1					
6	Уменьшение размера модуля (МОД)			14,7t			
7	Увеличение семантической связи (МОД)						
8	Уменьшение физической связанности (МОД)				1,47t		
9	Отложенное связывание (МОД)		2,94t				
10	Обнаружение атак (ИБ)						19,1t
11	Противостояние атакам (ИБ)						19,1t
12	Реакция на воздействие (ИБ)			7,3t			
13	Восстановление после атак (ИБ)			7,3t			

нальных точек для каждого конкретного проекта. Задача, таким образом, сводится к переборному нахождению такой комбинации базового шаблона архитектуры и тактик, при которых относительная величина трудоемкости является минимальной.

### 3. РЕЗУЛЬТАТЫ ИССЛЕДОВАНИЯ И ИХ ОБСУЖДЕНИЕ

В табл. 3 представлен пример относительной оценки шаблона архитектуры «Каналы и фильтры» для систем превентивного обслуживания.

Множество параметров качества для систем Интернета Вещей указанного типа будет

включать надежность, производительность, модифицируемость и информационную безопасность как первичные параметры. Особенностью выбранного шаблона архитектуры является высокий уровень надежности. При этом для достижения множества параметров качества необходимо реализовать дополнительно 11 тактик проектирования. Относительная стоимость реализации ( $n=10$ , а  $t_{\text{мод}}=0,5t$ ) указана в табл. 3.

Подставляя данные значения в выражение (3), получаем выражение для относительной оценки трудоемкости:

$$Q = \alpha \cdot EAF \cdot (n \cdot t + 104,93 \cdot t),$$

которое можно использовать в оценке и выборе результирующей архитектуры.

## ЗАКЛЮЧЕНИЕ

В настоящей статье был представлен метод оценки качества программной архитектуры систем Интернета Вещей на ранних стадиях проектирования. Приведен аналитический обзор, затрагивающий определение и классификацию существующих IoT-технологий, выделены основные значимые атрибуты качества систем IoT, а также приведены методы их оценки, влияющие на выбор программной архитектуры системы. Сформулирована аналитическая модель выбора программной архитектуры исходя из требуемых параметров качества системы при минимизации трудоемкости проекта. Метод на основе модели применен для оценки архитектуры для систем превентивного обслуживания. Дальнейшие направления исследований в данной области будут затрагивать обобщение данного метода на системы, выходящие за рамки области IoT, а также особенности применения метода на практике в процессе создания прототипов программных систем.

## СПИСОК ЛИТЕРАТУРЫ

1. Sundmaeker, H. et al. Vision and challenges for realising the Internet of Things // Cluster of European Research Projects on the Internet of Things, European Commission. – 2010. – Т. 3. – №. 3. – С. 34–36.
2. Hoepman, J. H. In things we trust? Towards trustability in the Internet of Things // International Joint Conference on Ambient Intelligence. – Springer, Berlin, Heidelberg, 2011. – С. 287–295.
3. Рекомендация МСЭ-Т Y.2060. Обзор интернета вещей. – Режим доступа: <https://www.itu.int/ITU-T/recommendations/rec.aspx?rec=11559&lang=ru>. – (дата обращения: 12.03.2019).

4. Internet of Things Awards. – Режим доступа: <http://postscapes.com/internet-of-things-award/2014/>. – (дата обращения: 11.03.2019).

5. Bassi, A. et al. Enabling things to talk. – Springer-Verlag GmbH, 2013.

6. ГОСТ Р ИСО/МЭК 7498-1-99 Информационная технология. Взаимосвязь открытых систем. Базовая эталонная модель. Часть 1. Базовая модель.

7. Dransfeld, H. ISG Provider Lens-Germany 2018 Internet of Things (I4. 0) Platforms, Services & Solutions. – 2017.

8. Gubbi, J. et al. Internet of Things (IoT): A vision, architectural elements, and future directions // Future generation computer systems. – 2013. – Т. 29. – №. 7. – С. 1645–1660.

9. Khan, R. et al. Future internet: the internet of things architecture, possible applications and key challenges // 2012 10th international conference on frontiers of information technology. – IEEE, 2012. – С. 257–260.

10. ГОСТ Р ИСО/МЭК 25010-2015 Информационные технологии. Системная и программная инженерия. Требования и оценка качества систем и программного обеспечения (SQuaRE). Модели качества систем и программных продуктов.

11. Garlan D., Shaw M. An introduction to software architecture // Advances in software engineering and knowledge engineering. – 1993. – С. 1–39.

12. Boehm, B. W. Software engineering economics // IEEE transactions on Software Engineering. – 1984. – №. 1. – С. 4–21.

13. Boehm, B. et al. Cost models for future software life cycle processes: COCOMO 2.0 // Annals of software engineering. – 1995. – Т. 1. – №. 1. – С. 57–94.

**Ядгарова Юлия Владимировна** – аспирант, Московский государственный технический университет им. Н. Э. Баумана, e-mail: [y.v.yadgarova@gmail.com](mailto:y.v.yadgarova@gmail.com)

**Таратухин Виктор Владимирович** – канд. техн. наук, профессор, Национальный исследовательский университет «Высшая школа экономики», e-mail: [victor.taratukhin@sap.com](mailto:victor.taratukhin@sap.com)

Ю. В. Ядгарова, В. В. Таратухин

## THE MODEL OF SOFTWARE ARCHITECTURAL PATTERN AND DESIGN TACTICS SELECTION FOR INTERNET OF THINGS SYSTEMS

Yu. V. Yadgarova\*, V. V. Taratukhin\*\*

*\*Bauman Moscow State Technical University*

*\*\*National Research University Higher School of Economics*

**Annotation.** At the current paper the analytical model and method of software architectural pattern and tactics selection for Internet of Things systems was presented. The meaning of IoT is generalized, vertical and horizontal classification of IoT systems has been provided with base quality attributes for such systems. Necessary software quality attributes were achieved by using base architectural pattern along with design tactics. Presented analytical model describes achievement of the software quality attributes with regards to project effort calculated with COCOMO II model. This approach allows to minimize effort calculation function and select suitable architectural patterns and design tactics on the earliest stages of the project. Usage of the presented methods potentially minimizes errors on the software architecture design stage, when only requirements and system prerequisites are available. As a result, the evaluation of one of the architectural patterns for IoT preventive maintenance system is presented. Proposed approach could be used on the earliest stages of the project for achieving the necessary quality attributes of the software and minimizing software cost as well as during prototype creation within strict time frame.

**Keywords:** Internet of Things, quality of IoT systems, IoT architecture, IoT architecture design, software quality attributes, design tactics, architectural patterns.

**Yadgarova Yulia** – PhD-student, Bauman Moscow State Technical University, e-mail: victor.taratukhin@sap.com

**Taratukhin Victor** – PhD, prof., National Research University Higher School of Economics, e-mail: victor.taratukhin@sap.com