УДК 004.273
DOI: 10.26102/2310-6018/2026.53.2.014

Количественная оценка архитектуры сложных программных систем на основе графовой многокритериальной модели

Я.Д. Саенко✉

Сибирский федеральный университет, Красноярск, Российская Федерация

Резюме. Работа посвящена исследованию количественной оценки архитектуры сложных программных систем, что является важной задачей для повышения надежности, производительности и масштабируемости. В современных методиках проектирования архитектур отсутствует формализованный и воспроизводимый способ системного анализа компонентов и их взаимодействий, что затрудняет сравнение альтернативных архитектурных решений и выявление наиболее эффективных структурных конфигураций на этапе проектирования. В связи с этим, данная статья направлена на разработку метода, позволяющего количественно оценивать архитектуру сложных программных систем с учетом значимости компонентов и их взаимодействий. Ведущим подходом к исследованию данной проблемы является графовое представление архитектуры, где вершины соответствуют программным компонентам с числовыми характеристиками по заранее заданным критериям качества, а ребра отражают архитектурные связи с коэффициентами влияния компонентов. Архитектурная значимость компонентов вычисляется как среднее значение коэффициентов входящих связей, при этом компоненты без входящих ребер. Итоговая оценка архитектуры определяется как взвешенное среднее локальных оценок компонентов с учетом их архитектурной значимости, что обеспечивает комплексный и системный подход к анализу архитектуры. В статье представлены результаты применения метода на примере программной системы с 10 и 13 компонентами, раскрыты изменения итоговой оценки при добавлении новых компонентов и изменении структуры связей, выявлены наиболее значимые с точки зрения архитектуры элементы системы. Полученные данные позволяют количественно сравнивать альтернативные архитектурные решения и выявлять влияние конкретных компонентов на эффективность всей системы. Материалы статьи представляют практическую ценность для проектирования, оптимизации и модернизации сложных программных систем, а также могут быть использованы в исследованиях в области инженерии программного обеспечения и системного анализа.

Ключевые слова: архитектура сложных программных систем, количественная оценка, графовая модель, многокритериальный анализ, архитектурная значимость.

Для цитирования: Саенко Я.Д. Количественная оценка архитектуры сложных программных систем на основе графовой многокритериальной модели. *Моделирование, оптимизация и информационные технологии.* 2026;14(2). URL: https://moitvivt.ru/ru/journal/article?id=2180 DOI: 10.26102/2310-6018/2026.53.2.014

Quantitative evaluation of the architecture of complex software systems based on a graph multi-criteria model

I.D. Saenko✉

Siberian Federal University, Krasnoyarsk, the Russian Federation

Abstract. The work explores the digital assessment of the structure of complex software systems, which is an important factor for improving reliability, performance, and scalability. Current design methods lack a formalized and reproducible method for architectural system analysis of components and their interactions, hindering the comparison of alternative architectural solutions and identifying the most effective structural configurations during the design phase. Therefore, this paper focuses on a development method that enables quantitative assessment of the architecture of complex software

systems, taking into account the implementation specifics of components and their interactions. The leading approach to studying this problem is a graph representation of the architecture, where the nodes correspond to software components with numerical characteristics according to pre-defined quality criteria, and the edges reflect architectural connections with component influence coefficients. The architectural significance of components is calculated as the average value of coefficients in incoming connections, while components without incoming connections. The final architectural score is defined as a weighted average of local component scores, taking into account their architectural significance, which provides a comprehensive and systematic approach to architectural analysis. The article presents the results of applying the method to a software system with 10 and 13 components, reveals changes in the final assessment when adding new components and changing the connection structure, and identifies the most significant elements of the system from an architectural perspective. The obtained data allows for a quantitative comparison of alternative architectural solutions and identifies the impact of components on the overall system's performance. The article's materials are of practical value for the design, optimization, and modernization of complex software systems, and can also be used in research in software engineering and systems analysis.

Keywords: architecture of complex software systems, quantitative assessment, graph model, multicriteria analysis, architectural significance.

For citation: Saenko I.D. Quantitative evaluation of the architecture of complex software systems based on a graph multi-criteria model. *Modeling, Optimization and Information Technology*. 2026;14(2). (In Russ.). URL: https://moitvivt.ru/ru/journal/article?id=2180 DOI: 10.26102/2310-6018/2026.53.2.014

Введение

Архитектура программной системы является ключевым фактором, определяющим ее качество, масштабируемость и устойчивость к изменениям [1]. В условиях роста сложности программных решений возникает необходимость в формализованных методах оценки архитектуры, позволяющих объективно сравнивать архитектурные альтернативы и анализировать последствия архитектурных изменений, в соответствии с требованиями к архитектуре программных систем (ГОСТ Р 57100-2025[1]).

Существующие подходы к оценке архитектуры, такие как Architecture Tradeoff Analysis Method (ATAM)[2] [2] и Cost Benefit Analysis Method (CBAM)[3], широко используются на практике [3]. Однако данные методы ориентированы преимущественно на качественный анализ и сценарное моделирование, что затрудняет получение воспроизводимых количественных показателей. Кроме того, применение ATAM и CBAM связано с высокой трудоемкостью, требует значительных временных ресурсов и плохо масштабируется при увеличении числа компонентов и архитектурных вариантов. Это ограничивает возможность их использования в задачах автоматизированного анализа и при частых изменениях архитектуры в ходе эволюции системы [1, 4].

Таким образом, задача проектирования архитектуры сложных программных систем (СПС) и существующие методики показывают отсутствие возможности анализа архитектуры без учета сценарного воспроизведения, в облегченной, статической форме, без обработки потока данных. В связи с этим, в данной работе предлагается метод количественной оценки архитектуры сложных программных систем, основанный на многокритериальном анализе компонентов [4, 5] и учете структуры их взаимосвязей в

[1] ГОСТ Р 57100-2025. Системная и программная инженерия. Описание архитектуры: национальный стандарт Российской Федерации: издание официальное: утвержден и введен в действие Приказом Федерального агентства по техническому регулированию и метрологии от 25 марта 2025 г. № 215-ст: введен взамен ГОСТ Р 57100-2016/ISO/IEC/IEEE 42010:2011: дата введения 2025-06-30. Москва: Российский институт стандартизации; 2025. 27 с.
[2] Kazman R., Klein M., Clements P. *ATAM: Method for Architecture Evaluation*. Technical Report CMU/SEI-2000-TR-004. Pittsburgh: Software Engineering Institute, Carnegie Mellon University; 2000. 70 p.
[3] Nord R., Barbacci M.R., Clements P.C., et al. *Integrating the Architecture Tradeoff Analysis Method (ATAM) with the Cost Benefit Analysis Method (CBAM)*. Technical Note CMU/SEI-2003-TN-038. Pittsburgh: Carnegie Mellon University; 2003. 24 p. https://doi.org/10.1184/R1/6574613.v1

виде графовой [6] модели. Предлагаемый подход позволяет формировать интегральные показатели качества архитектуры и анализировать влияние структурных изменений, таких как добавление или удаление компонентов, на итоговую оценку системы.

Материалы и методы

Общая модель проектирования архитектуры сложной программной системы может быть представлена в виде функциональной модели IDEF0[4], отражающей последовательность и взаимосвязь основных этапов архитектурного проектирования. В рамках данной модели входом процесса являются функциональные и нефункциональные требования к системе, а также ограничения предметной области и среды эксплуатации. Управляющими воздействиями выступают архитектурные принципы, стандарты проектирования и выбранные критерии качества архитектуры. В качестве механизмов используются методы архитектурного анализа, средства моделирования и экспертные знания проектировщиков. Выходом модели является формализованное архитектурное описание программной системы, включающее состав компонентов, их характеристики и структуру взаимодействий.

Функционально процесс проектирования архитектуры СПС декомпозируется на несколько взаимосвязанных подпроцессов. На первом этапе осуществляется идентификация и декомпозиция программных компонентов на основе требований и функций системы. Далее выполняется формирование архитектурных связей между компонентами с учетом потоков данных и управления. На следующем этапе производится оценка компонентов по заданным критериям качества и определение их архитектурной значимости в структуре системы. Завершающий этап предполагает интеграцию полученных оценок и формирование целостной архитектурной модели, пригодной для количественного анализа и сравнения альтернативных архитектурных решений. Представление процесса в нотации IDEF0, продемонстрированный на Рисунке 1, обеспечивает наглядность, формализацию и воспроизводимость проектных решений, а также создает основу для последующей автоматизации анализа архитектуры программных систем. Что, в свою очередь, играет решающую роль в поддержке принятия решений [7] для автоматизации процесса проектирования архитектуры СПС [8].

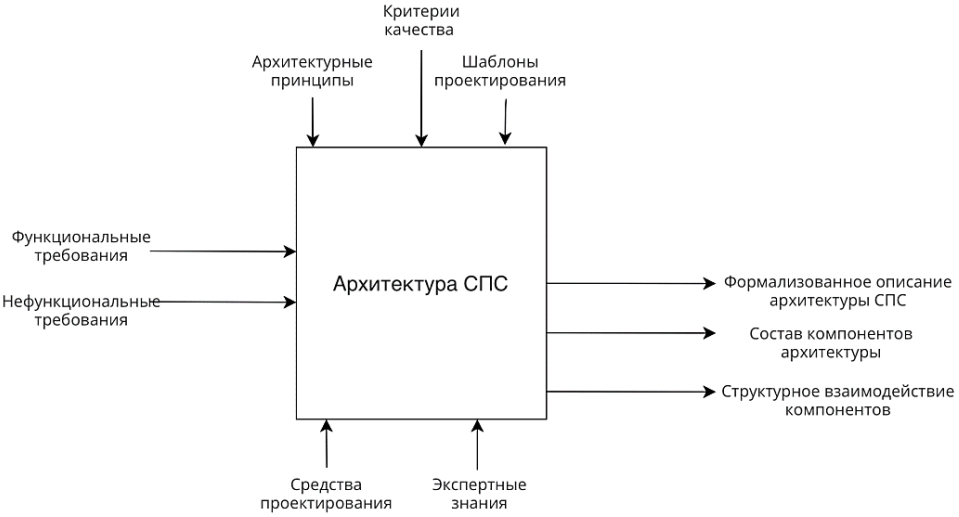

Рисунок 1 – Представление архитектуры СПС в IDEF0 нотации
Figure 1 – Representation of the CSS architecture in IDEF0 notation

[4] Кинзябулатов Р. Перевод стандарта IDEF0 на русский язык. Кинзябулатов Рамиль Хибатуллович. Компания Тринион. URL: https://trinion.org/blog/perevod-standarta-idef0-s-angliyskogo-na-russkiy-yazyk (дата обращения: 26.01.2026).

Для количественной оценки архитектуры программной системы вводится набор критериев качества, отражающих ключевые характеристики архитектурных решений:

$$C = \{c_1, c_2, \dots, c_k\}, \quad (1)$$

каждому из которых сопоставлен весовой коэффициент w_i, характеризующий относительную важность соответствующего критерия. Весовые коэффициенты нормированы таким образом, что их сумма равна единице:

$$\sum_{i=1}^{k} w_i = 1. \quad (2)$$

В качестве критериев могут использоваться такие показатели, как надежность, производительность, масштабируемость, сопровождаемость и другие характеристики, значимые для рассматриваемой архитектуры.

Архитектура программной системы формализуется в виде ориентированного графа [6]:

$$G = (V, E), \quad (3)$$

где V – множество программных компонентов, E – множество архитектурных связей между компонентами.

Каждой вершине $v \in V$ сопоставляется вектор числовых оценок по заданным критериям качества [5]:

$$P(v) = \{p_{v,1}, p_{v,2}, \dots, p_{v,k}\}, \quad (4)$$

где значения $p_{v,i} \in [0,10]$ отражают степень соответствия компонента v критерию c_i. Таким образом, графовая модель позволяет одновременно учитывать как свойство отдельных компонентов, так и структуру их взаимодействий.

Количественная оценка архитектуры начинается с вычисления локальной оценки каждого компонента. Локальная оценка определяется как агрегированное значение его характеристик по всем выбранным критериям[5] и вычисляется в виду взвешенной суммы:

$$S(v) = \sum_{i=1}^{k} w_i \cdot p_{v,i}. \quad (5)$$

В частном случае, когда все критерии считаются равнозначными, локальная оценка компонента может быть вычислена как среднее арифметическое значений его критериев.

Для учета влияния взаимодействий между компонентами каждой архитектурной связи $e = (u, v) \in E$ сопоставляется коэффициент влияния:

$$q(u, v) \in [0,1], \quad (6)$$

характеризующий степень архитектурной зависимости компонента *v* от компонента *u*. Значения коэффициентов могут определяться на основе типа взаимодействия, интенсивности обмена данными либо с использованием экспертных оценок.

С целью отражения структурной роли компонента в архитектуре [9, 10] вводится показатель архитектурной значимости компонента. Архитектурная значимость $\lambda(v)$ определяется на основе входящих связей и вычисляется следующим образом:

$$\lambda(v) = \begin{cases} 1, & |\ N(v)\ | = 0 \\ \frac{\sum_{u \in N(v)} q(u,v)}{|N(v)|}, & |\ N(v)\ | > 0 \end{cases}, \quad (7)$$

[5] Подиновский В.В. *Многокритериальные задачи принятия решений: теория и методы анализа*. Москва: Издательство Юрайт; 2023. 486 с.

где $N(v)$ – множество компонентов, имеющих входящих в компонент v. Такой подход позволяет учитывать как изолированные компоненты, так и элементы, играющие ключевую роль в структуре системы.

Итоговая количественная оценка архитектуры вычисляется как взвешенное среднее локальных оценок компонентов:

$$A = \frac{\sum_{v \in V} S(v) \cdot \lambda(v)}{\sum_{v \in V} \lambda(v)}. \quad (8)$$

Тем самым, вклад каждого компонента в итоговую оценку архитектуры определяется не только его собственными характеристиками, но и его положением и ролью в общей структуре архитектуры программной системы.

Таким образом, задача проектирования архитектуры программной системы в рамках предлагаемого подхода может быть сформулирована как задача оптимизации. При фиксированных требованиях, наборе критериев качества и ограничениях предметной области необходимо выбрать состав программных компонентов и структуру архитектурных связей между ними таким образом, чтобы итоговая количественная оценка архитектуры A принимала максимальное возможное значение.

Формально задача сводится к максимизации целевой функции:

$$A \to max. \quad (9)$$

Такая постановка задачи позволяет рассматривать проектирование архитектуры как формализованный процесс поиска оптимальной структурной конфигурации, а также создает основу для сравнения альтернативных архитектурных решений и применения методов автоматизированного анализа и оптимизации.

В рамках экспериментального исследования рассматривается архитектура программной системы, состоящая из десяти компонентов. Наглядное отображение структуры архитектуры приведено на Рисунке 2; значения метрик по каждому компоненту приведены в Таблице 1, а значения связей между компонентами – в Таблице 2. Для количественной оценки архитектуры используются три типовых унифицированных критерия качества для различного рода сложных программных систем: надежность, производительность и масштабируемость, при этом все критерии считаются равнозначными и имеют одинаковые весовые коэффициенты.

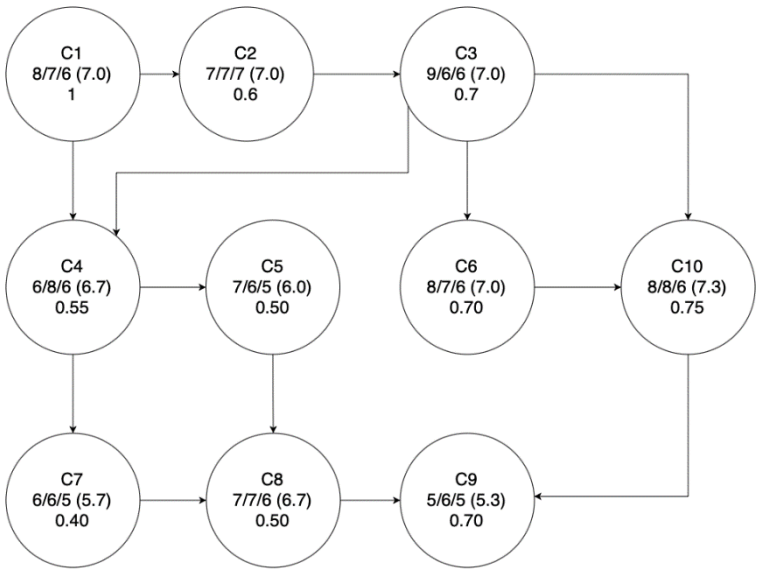

Рисунок 2 – Архитектура программной системы из 10 компонентов
Figure 2 – Architecture of a software system consisting of 10 components

Таблица 1 – Значения критериев компонентов
Table 1 – Values of component criteria

Компонент	Надежность	Производительность	Масштабируемость	$S(v)$
C1	8	7	6	7,0
C2	7	7	7	7,0
C3	9	6	6	7,0
C4	6	8	6	6,7
C5	7	6	5	6,0
C6	8	7	6	7,0
C7	6	6	5	5,7
C8	7	7	6	6,7
C9	5	6	5	5,3
C10	8	8	6	7,3

Таблица 2 – Коэффициенты влияния связей
Table 2 – Coefficients of influence of connections

№	$u \to v$	$q(u,v)$
1	C1→C2	0,6
2	C1→C4	0,5
3	C2→C3	0,7
4	C3→C10	0,8
5	C3→C4	0,6
6	C3→C6	0,7
7	C4→C5	0,5
8	C4→C7	0,4
9	C5→C8	0,6
10	C6→C10	0,7
11	C7→C8	0,4
12	C8→C9	0,5
13	C10→C9	0,9

На основе коэффициентов влияния архитектурных связей между компонентами осуществляется вычисление показателей их архитектурной значимости, отражающих структурную роль элементов системы [9, 10], значения в Таблице 3.

Таблица 3 – Архитектурная значимость компонентов
Table 3 – Architectural significance of components

Компонент	$\lambda(v)$
C1	1,00
C2	0,60
C3	0,70
C4	0,55
C5	0,50
C6	0,70
C7	0,40
C8	0,50
C9	0,70
C10	0,75

Итоговая оценка архитектуры:

$$\sum S(v)\lambda(v) = 7{,}0 \cdot 1{,}0 + 7{,}0 \cdot 0{,}6 + 7{,}0 \cdot 0{,}7 + 6{,}7 \cdot 0{,}55 + 6{,}0 \cdot 0{,}5 + 7{,}0 \cdot 0{,}7 + \\ + 5{,}7 \cdot 0{,}4 + 6{,}7 \cdot 0{,}5 + 5{,}3 \cdot 0{,}7 + 7{,}3 \cdot 0{,}75 = 42{,}5. \quad (10)$$

$$\sum \lambda(v) = 1{,}0 + 0{,}6 + 0{,}7 + 0{,}55 + 0{,}5 + 0{,}7 + 0{,}4 + 0{,}5 + 0{,}7 + 0{,}75 = 6{,}4. \quad (11)$$

$$A_{10} = \frac{42{,}50}{6{,}40} \approx 6{,}64. \quad (12)$$

Рисунок 2 демонстрирует компонент C7 как потенциальную точку для улучшения архитектуры системы, поскольку его текущая локальная оценка составляет 5,7 при архитектурной значимости 0,4. Такие показатели указывают на то, что C7 имеет средний уровень качества по выбранным критериям и относительно высокое влияние на структуру системы. В связи с этим целенаправленная оптимизация характеристик данного компонента и его взаимодействий с другими элементами архитектуры способна существенно повысить итоговую оценку системы. Проведение изменений в компоненте C7 позволит улучшить как его собственные показатели по критериям качества, так и интегральные свойства всей архитектуры, что подтверждает необходимость включения данного этапа в процесс проектирования и модернизации программной системы.

Предлагается расширение архитектуры (Рисунок 3) за счет добавления новых специализированных компонентов, а также перераспределение архитектурных связей с учетом коэффициентов влияния. Для каждого компонента рассчитываются целевые показатели по выбранным критериям качества (Таблицы 4–6), что позволяет определить приоритеты улучшений и их ожидаемый эффект на систему. Такие корректировки направлены на повышение локальных оценок отдельных компонентов, увеличение их архитектурной значимости и, как следствие, улучшение итоговой количественной оценки архитектуры. Реализация этих изменений обеспечивает системный подход к оптимизации структуры программной системы и позволяет объективно оценивать влияние каждого внесенного элемента на эффективность и устойчивость всей архитектуры.

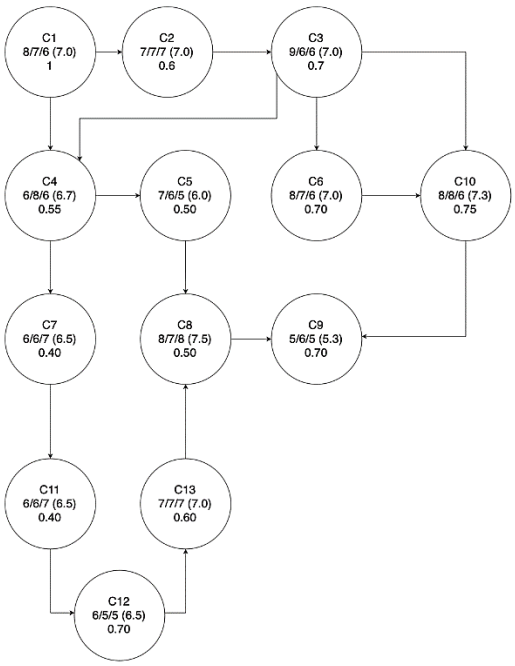

Рисунок 3 – Архитектура программной системы из 13 компонентов
Figure 3 – Architecture of a software system consisting of 13 components

Таблица 4 – Архитектурная значимость компонентов
Table 4 – Architectural significance of components

Компонент	Надежность	Производительность	Масштабируемость	$S(v)$
C7	6	6	7	6,5
C8	8	7	8	7,5
C11	6	6	7	6,5
C12	6	5	5	6,5
C13	7	7	7	7,0

Таблица 5 – Архитектурная значимость компонентов
Table 5 – Architectural significance of components

Связь	$q(u,v)$
C7 → C11	0,4
C11 → C10	0,7
C12 → C11	0,6
C13 → C13	0,5

Таблица 6 – Архитектурная значимость компонентов
Table 6 – Architectural significance of components

Компонент	$\lambda(v)$
C7	0,40
C8	0,50
C11	0,40
C12	0,70
C13	0,60

Итоговая оценка архитектуры:

$$\sum S(v) \cdot \lambda(v) \approx 56{,}1. \quad (13)$$

$$\sum \lambda(v) = 8{,}1. \quad (14)$$

$$A_{13} = \frac{56{,}1}{8{,}15} \approx 6{,}88. \quad (15)$$

Результаты и обсуждение

Разработанный метод демонстрирует, что изменение структуры архитектуры СПС и перераспределение связей между компонентами оказывает значительное влияние на интегральную оценку архитектуры программной системы. В частности, добавление новых компонентов изменяет распределение архитектурной значимости $\lambda(v)$, что влияет на весовой вклад каждого компонента в итоговую оценку. Это подтверждает необходимость учитывать не только локальные характеристики компонентов, но и их положение в структуре системы, а также степень взаимодействия с другими элементами. Таким образом, даже небольшие изменения в топологии архитектуры могут привести к существенным изменениям в интегральной оценке, что делает графовое представление и расчет архитектурной значимости ключевыми инструментами анализа. Более того, в представленной Таблице 7 видно, что у архитектурной альтернативы 2 суммарная оценка выше, из чего можно сделать вывод, что исходя из первоначально поставленной задачи по достижению наибольшей суммарной оценки, она более предпочтительна.

Таблица 7 – Архитектурная значимость компонентов
Table 7 – Architectural significance of components

№ архитектурной альтернативы	Количество компонентов	Суммарная оценка
1	10	6,64
2	13	6,88

Добавление специализированных компонентов позволяет перераспределить архитектурную значимость в пользу элементов с более высокими локальными характеристиками, повышая общую эффективность архитектуры. Такой эффект демонстрирует, что целенаправленное проектирование и расширение системы с учетом функциональных связей позволяет не только повысить интегральную оценку, но и выявить критически важные компоненты, которые оказывают наибольшее влияние на системные показатели. В этом контексте методика количественного анализа архитектуры становится инструментом для принятия обоснованных инженерных решений при разработке новых систем или модернизации существующих.

Кроме того, проведенный анализ показывает, что ориентация ребер и коэффициенты влияния между компонентами играют важную роль в распределении архитектурной значимости. Компоненты с большим числом входящих связей или сильными влияющими связями получают более высокий вес в итоговой оценке, что подчеркивает необходимость тщательной проработки взаимодействий при проектировании архитектуры. Это позволяет не только выявлять наиболее значимые элементы, но и прогнозировать последствия структурных изменений для общей эффективности системы, обеспечивая более системный и научно обоснованный подход к архитектурному проектированию.

Заключение

Рассмотренный метод отличается от существующих подходов к оценке архитектуры тем, что обеспечивает прямую количественную оценку без необходимости моделирования сценариев использования системы. В отличие от трудоемких методов, требующих значительных объемов данных о функционировании и нагрузках, разработанная методика позволяет оперативно получить интегральную оценку архитектуры, учитывающую как локальные характеристики компонентов, так и структуру их взаимосвязей. Это делает анализ архитектуры более наглядным и удобным для сравнения альтернативных проектных решений и выявления критически значимых компонентов.

В ходе работы разработан и апробирован метод количественной оценки архитектуры сложных программных систем, основанный на графовой многокритериальной модели. Метод позволяет вычислять архитектурную значимость компонентов и интегральную оценку архитектуры, обеспечивая воспроизводимый и интерпретируемый подход к анализу. Экспериментальные результаты показали, что изменение структуры архитектуры и добавление новых компонентов существенно влияют на итоговую оценку, что подтверждает практическую применимость метода при проектировании и модернизации программных систем.

Постановка задачи проектирования архитектуры в виде задачи максимизации интегральной оценки позволяет рассматривать процесс как задачу оптимизации, направленную на выбор наилучшей конфигурации компонентов и связей. В качестве направления дальнейших исследований предлагается интеграция сетей Петри для

моделирования динамических взаимодействий между компонентами. Это позволит учитывать параллельность и синхронизацию процессов, выявлять узкие места и повышать точность количественной оценки архитектуры.

СПИСОК ИСТОЧНИКОВ / REFERENCES

1. Bass L., Clements P., Kazman R. *Software Architecture in Practice*. Addison-Wesley Professional; 2021. 464 p.
2. Maheshwari P., Teoh A. Supporting ATAM with a collaborative Web-based software architecture evaluation tool. *Science of Computer Programming*. 2005;57(1):109–128. https://doi.org/10.1016/j.scico.2004.10.008
3. Корниенко Д.В., Никулин А.В. Анализ методов оценки архитектуры программного обеспечения. В сборнике: *Технологии и техника: пути инновационного развития: Сборник научных статей 3-й Международной научно-технической конференции, 17 июня 2025 года, Воронеж, Россия.* Курск: Университетская книга; 2025. С. 146–153.
Kornienko D.V., Nikulin A.V. Analysis of software architecture assessment methods. In: *Tekhnologii i tekhnika: puti innovatsionnogo razvitiya: Sbornik nauchnykh statei 3-i Mezhdunarodnoi nauchno-tekhnicheskoi konferentsii, 17 June 2025, Voronezh, Russia.* Kursk: Universitetskaya kniga; 2025. P. 146–153. (In Russ.).
4. Ashraf M.U., Aljedaibi W. ATAM-based Architecture Evaluation Using LOTOS Formal Method. *International Journal of Information Technology and Computer Science.* 2017;9(3):10–18. https://doi.org/10.5815/ijitcs.2017.03.02
5. Ксенофонтова Е.А. Многокритериальный анализ. *Проблемы науки.* 2020;(11):30–31.
6. Игнацкая И.В., Лукин В.Н. Моделирование программных систем на основе графа взаимодействий. *Вестник Московского авиационного института*. 2009;16(7):70–75.
7. Симанков В.С., Халафян А.А. Системный подход к разработке медицинских систем поддержки принятия решений. *Известия высших учебных заведений. Северо-Кавказский регион. Технические науки*. 2010;(1):29–36.
Simankov V.S., Khalaphyan A.A. The system approach to working out of medical decision support system. *Bulletin of Higher Educational Institutions. North Caucasus Region. Technical Sciences*. 2010;(1):29–36. (In Russ.).
8. Кузнецов А.С., Ченцов С.В., Царев Р.Ю. *Многоэтапный анализ архитектурной надежности и синтез отказоустойчивого программного обеспечения сложных систем.* Красноярск: Сибирский федеральный университет; 2013. 142 с.
9. Рындин А.А., Шитиков Д.В. Применение компонентно-ориентированного подхода при разработке информационных систем с микросервисной архитектурой. *Вестник Воронежского государственного технического университета*. 2022;18(6):15–20. https://doi.org/10.36622/VSTU.2022.18.6.002
Ryndin A.A., Shitikov D.V. Application of a component-based software engineering in the development of information systems with microservice architecture. *Bulletin of Voronezh State Technical University.* 2022;18(6):15–20. (In Russ.). https://doi.org/10.36622/VSTU.2022.18.6.002
10. Saaty Th.L. Decision Making with the Analytic Hierarchy Process. *International Journal of Services Sciences*. 2008;1(1):83–98. https://doi.org/10.1504/IJSSCI.2008.017590

ИНФОРМАЦИЯ ОБ АВТОРЕ / INFORMATION ABOUT THE AUTHOR

Саенко Ярослав Дмитриевич, аспирант, Сибирский федеральный университет, Красноярск, Российская Федерация.
e-mail: saenkoyd@ya.ru
ORCID: 0009-0005-2580-5352

Iaroslav D. Saenko, Postgraduate, Siberian Federal University, Krasnoyarsk, the Russian Federation.

Статья поступила в редакцию 22.01.2026; одобрена после рецензирования 16.02.2026; принята к публикации 27.02.2026.

The article was submitted 22.01.2026; approved after reviewing 16.02.2026; accepted for publication 27.02.2026.