|  |
| --- |
| https://lh6.googleusercontent.com/QcftzNtI05T0Y6fjdSh1Rr2rt8oqZ1IvnLvbn1jLJ7CCyteVir3k-xBLv4SL1wAgWJsRhmmJSR0UW-RP63_GQenE4vVWv05BRoZTsmIcBccVTnfxwmsnNMvjg599x9SqZd8E3dkd |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждениевысшего образования«МИРЭА - Российский технологический университет»РТУ МИРЭА |

Институт Информационных Технологий (ИТ)

Кафедра инструментального и прикладного программного обеспечения

(ИиППО)

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №3** | |
| **по дисциплине** | |
| «Архитектура программных продуктов и систем» | |
|  | |
| Выполнил студент группы ИКБО-13-17 | Шатилов А.А. |
| Принял | Куликов А.А. |

|  |  |  |
| --- | --- | --- |
| Лабораторные работы выполнены | «\_\_»\_\_\_\_\_\_\_202\_\_ г. | *(подпись студента)* |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_202\_\_ г. | *(подпись руководителя)* |

**Цель работы**

Цель данной практической работы заключается в получении навыков оценки

и анализа архитектур программных систем.

**Задание:**

Из приведённого в методичке списка методов анализа дизайна архитектуры необходимо выбрать 2 метода (выбираете самостоятельно, можно выбирать в том числе и отсутствующие в этом списке), дать их подробное описание (кем разработан, когда разработан, преимущества, недостатки и т.д.) и провести сравнительный анализ выбранных методов.

**Ход работы**

В этой работе будут рассмотрены следующие подходы анализа дизайна архитектуры: *Software Architecture Analysis Method* и *Architecture Tradeoff Analysis Method.*

Обе методики оценивают качество модифицируемости и могут применяться к любым типам систем.

*Software Architecture Analysis Method*

Метод анализа архитектуры программного обеспечения (Software Architecture Analysis Method, SAAM) — это метод, используемый в архитектуре программного обеспечения для её оценки. Это был первый документированный метод анализа архитектуры программного обеспечения, который был разработан в середине 1990-х годов для анализа системы на возможность модификации; полезен для тестирования любого нефункционального аспекта ПО.

SAAM позже был расширен для анализа архитектуры относительно показателей качества, таких как модифицируемость, портируемость, расширяемость, интегрируемость и функциональный охват.

Плюсы подхода:

* Определение потенциальных областей высокой сложности.
* Открыт для любого архитектурного описания.

Недостатки подхода:

* Нет явной метрики качества.
* Не отражает шаги проектирования/разработки.

Основные метрики:

* Классификация сценариев.

Основные его шаги:

1. Определить набор сценариев действий пользователей или внешних систем, использующих некоторые возможности, которые могут уже планироваться для реализации в системе или быть новыми. Сценарии должны быть значимы для конкретных заинтересованных лиц, будь то пользователь, разработчик, ответственный за сопровождение, представитель контролирующей организации и пр. Чем полнее набор сценариев, тем выше будет качество анализа. Можно также оценить частоту появления и важность сценариев, возможный ущерб от невозможности их выполнить.
2. Определить архитектуру (или несколько сравниваемых архитектур). Это должно быть сделано в понятной всем участникам оценки форме.
3. Классифицировать сценарии. Для каждого сценария из набора должно быть определено, поддерживается ли он уже данной архитектурой или для его поддержки нужно вносить в нее изменения. Сценарий может поддерживаться, т.е. его выполнение не потребует внесения изменений ни в один из компонентов, или же не поддерживаться, если его выполнение требует изменений в описании поведения одного или нескольких компонентов, или изменений в их интерфейсах. Поддержка сценария означает, что лицо, заинтересованное в его выполнении, оценивает степень поддержки как достаточную, а необходимые при этом действия как достаточно удобные.
4. Оценить сценарии. Определить, какие из сценариев полностью поддерживаются рассматриваемыми архитектурами. Для каждого неподдерживаемого сценария надо определить необходимые изменения в архитектуре — внесение новых компонентов, изменения в существующих, изменения связей и способов взаимодействия. Если есть возможность, стоит оценить трудоемкость внесения таких изменений.
5. Выявить взаимодействие сценариев. Определить какие компоненты требуется изменять для неподдерживаемых сценариев; компоненты, которые требуется изменять для поддержки нескольких сценариев — такие сценарии называют взаимодействующими; оценить степень смысловой связанности взаимодействующих сценариев. Малая связанность по смыслу между взаимодействующими сценариями означает, что компоненты, в которых они взаимодействуют, выполняют слабо связанные между собой задачи и их стоит декомпозировать. Компоненты, в которых взаимодействуют много (> 2-х) сценариев, также являются возможными проблемными местами.
6. Оценить архитектуру в целом (или сравнить несколько заданных архитектур). Для этого надо использовать оценки важности сценариев и степень их поддержки архитектурой.

*Architecture Tradeoff Analysis Method*

Метод анализа компромисса архитектуры (Architecture Tradeoff Analysis Method, ATAM) представляет собой процесс, используемый на ранних этапах жизненного цикла разработки для снижения рисков. Является наследником метода анализа архитектуры ПО (Software Architecture Analysis Method, SAAM).

Был разработан Институтом Программной Инженерии университета Карнеги-Меллона. Основная цель – помощь в выборе подходящей архитектуры за счёт обнаружения компромиссов и слабых мест программной системы.

Лучше подходит при применении на ранних стадиях жизненного цикла разработки приложения, когда цена смены архитектуры минимальна.

Плюсы подхода:

* Выявление рисков на ранних стадиях разработки;
* Улучшенная взаимосвязь между заинтересованными сторонами;
* Определяются более точные требования к качеству;
* Более качественная архитектурная документация;
* Задокументированные обоснования для архитектурных решений.

Недостатки подхода:

* Требует глубоких технических знаний.
* Сложность донесения информации до заказчика.

Основные метрики:

* Точки чувствительности.
* Компромиссные точки.

ATAM формально состоит из девяти шагов:

1. Представить ATAM - представить концепцию ATAM заинтересованным сторонам и ответить на любые вопросы о процессе.
2. Существующие бизнес-драйверы - каждый участник процесса представляет и оценивает бизнес-драйверы для рассматриваемой системы.
3. Представить архитектуру - архитектор представляет команде высокоуровневую архитектуру с «соответствующим уровнем детализации».
4. Определить архитектурные подходы - команда представляет и обсуждает различные архитектурные подходы к системе.
5. Сгенерировать служебное дерево атрибутов качества - определите основные бизнес-требования и технические требования к системе и сопоставьте их с соответствующим архитектурным свойством. Представьте сценарий для данного требования.
6. Проанализировать архитектурные подходы - анализируйте каждый сценарий, оценивая их по приоритету. Затем архитектура оценивается по каждому сценарию.
7. Проведение мозгового штурма и определить приоритеты сценариев - среди более широкой группы заинтересованных сторон представить текущие сценарии и расширить их.
8. Анализировать архитектурные подходы - повтор шага 6 еще раз с дополнительными знаниями более широкого круга заинтересованных сторон.
9. Представление результатов - предоставить всю документацию заинтересованным сторонам.

Эти шаги разделены на две фазы:

1. Этап 1 состоит из шагов 1-6, и после этого этапа становятся известны состояние и контекст проекта, основные архитектурные требования и состояние архитектурной документации.
2. Этап 2 состоит из шагов 7–9 и завершает анализ и оценку решений.

*Сравнительный анализ: SAAM и ATAM*

В таблице 1 показана сравнительная таблица описываемых в текущей работе методов анализа дизайна архитектуры:

Таблица 1. Сравнение SAAM и ATAM

|  |  |  |
| --- | --- | --- |
| **Метод** | SAAM | ATAM |
| **Плюсы** | * Определение потенциальных областей высокой сложности. * Открыт для любого архитектурного описания. | * Выявление рисков на ранних стадиях разработки; * Улучшенная взаимосвязь между заинтересованными сторонами; * Определяются более точные требования к качеству; * Более качественная архитектурная документация; * Задокументированные обоснования для архитектурных решений. |
| **Минусы** | * Нет явной метрики качества. * Не отражает шаги проектирования/разработки. | * Требует глубоких технических знаний. * Сложность донесения информации до заказчика |

Таким образом можно сделать вывод, что SAAM подходит в тех случаях, когда нет возможности обеспечить глубокие технические знания, и при этом не требуется явных метрик качества и отражения этапов проектирования; в это же время ATAM походит, когда требуется более глубокий и технический анализ архитектуры.

**Заключение**

В ходе выполнения данной практической работы были изучены и сравнены два метода анализа дизайна архитектуры.

Были приобретены навыки анализа и выбора подходящих методов анализа дизайна архитектур программных проектов.

**Список литературы**

1. Software architecture analysis method // Wikipedia URL: <https://en.wikipedia.org/wiki/Software_architecture_analysis_method> (дата обращения: 02.03.2021).
2. Architecture tradeoff analysis method // Wikipedia URL: <https://en.wikipedia.org/wiki/Architecture_tradeoff_analysis_method> (дата обращения: 02.03.2021).
3. Scenario-Based Software Architecture Evaluation Software Architecture Evaluation Methods: An Overview // Программная инженерия и анализ данных URL: <http://www.pps-design.org/mdd:overview> (дата обращения: 02.03.2021).