1 Анализ систем тестирования знаний языков программирования

Работа, проведенная в рамках исследовательской части позволяет подойти к решению описанных во введении проблем посредством формирования функциональных требований и составления диаграммы вариантов использования подсистемы тестирования знаний языков описания аппаратуры, полученных на основе анализа имеющихся систем тестирования знаний языков программирования и методов тестирования знаний, используемых в этих системах.

1.1 Классификация методов тестирования знаний

Перед функциональным моделированием различных методов тестирования знаний, необходимо ввести их классификацию.

В качестве основы была взята подобная классификация для системы дистанционного обучения (далее — СДО) Moodle [2]. Она была дополнена с учетом функциональных особенностей таких СДО, как Huawei University, Coursera, Stepik, Ethernaut, а также хакатона Paradigm CTF [3]. Сформированная классификация приведена в таблице 1.

Таблица 1 — Классификация методов тестирования знаний

№	Тип	Подтип
1	Тестирование с ответом в закрытой форме	1.1 Выбор одного ответа 1.2 Выбор множественных ответов 1.3 Сопоставление
2	Тестирование с коротким ответом	2.1 С автоматизированной проверкой 2.2 С проверкой преподавателем 2.3 С перекрестной проверкой
3	Тестирование с ответом в форме эссе	3.1 С проверкой преподавателем 3.2 С перекрестной проверкой

Продолжение таблицы 1

4	Тестирование на написание исходного кода	4.1 С проверкой по референсным значениям 4.2 Автоматизированное тестирование на проверяющей стороне 4.3 Другие

1.2 Методы тестирования знаний

1.2.1 Тестирование с ответом в закрытой форме

Тестирование с ответом в закрытой форме применяется практически во всех системах тестирования знаний. Соответствующая IDEF0-модель представлена на рисунке 1.

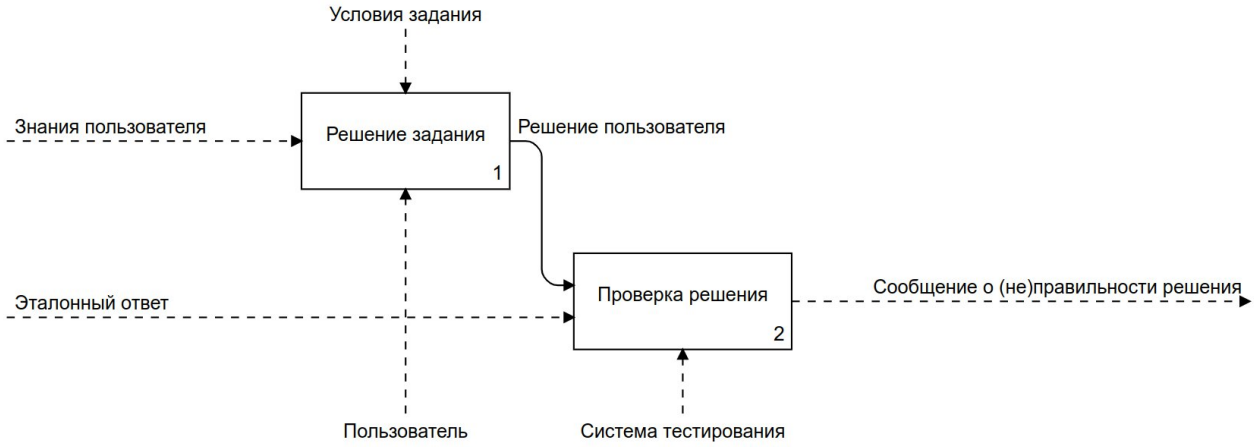

Рисунок 1 — Функциональная модель тестирования с ответом в закрытой форме

Основным недостатком такой реализации тестирования с ответом в закрытой форме является невозможность получить содержательную обратную связь в случае неверного решения. Возможные формы обратной связи для различных подтипов заданий с закрытым ответом показаны в таблице 2.

Таблица 2 — Формы обратной связи для тестирования с ответом в закрытой форме

Подтип тестирования	**Форма обратной связи**
Выбор одного ответа	Пояснение причин некорректности ответа
Выбор множественных ответов	Сообщение о выборе избыточного/недостаточного числа вариантов
Сопоставление	Сообщение о количестве неправильно выбранных пар
	Подсветка некорректно выбранных пар

В случае заданий на выбор множественных ответов сообщение о выборе избыточного/недостаточного числа вариантов неинформативно и не позволяет пользователю повторно проанализировать задание с его учетом. При этом такой вид обратной связи позволяет пользователю сократить число вариантов для перебора ответов при повторном решении задания. По этим причинам использование обратной связи в заданиях этого подтипа не всегда желательно.

В случае заданий на сопоставление сообщение о количестве неправильно выбранных пар менее информативно, но не сокращает число вариантов перебора ответа. Подсветка некорректно выбранных пар содержит полезную для повторного анализа задания информацию, но сокращает число вариантов перебора ответа.

Еще одной проблемой тестирования с ответом в закрытой форме, уже затронутой выше, является проблема перебора ответов. Данная проблема не возникает в случае проведения контрольных мероприятий, где количество попыток прохождения тестирования ограничено. Однако, в случае открытых онлайн-курсов, число попыток прохождения тестирования, как правило, не ограничивается. В таком случае одним из решений проблемы является ограничение времени до возможности повторно пройти тестирование.

– наглядность оценки (хотя процесс тестирования проходит в режиме черного ящика, результат тестирования, при должной подготовки системы автоматизированного и самих тестов, нагляден и способен в некоторой степени оповестить об ошибках в тестируемых приложениях);
– возможность применения для большого числа пользователей.

Основным недостатком такого подхода является сложность его реализации.

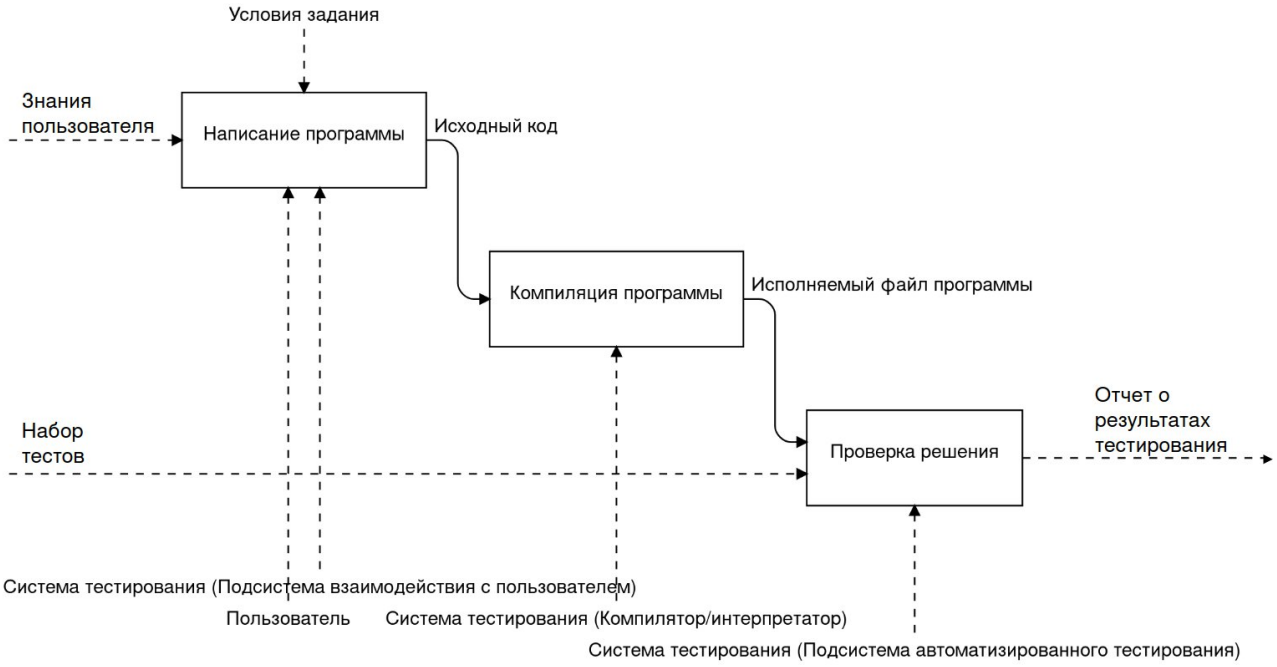

Рисунок 7 — Функциональная модель тестирования на написание программы с автоматизированной проверкой

1.3 Функциональные требования и диаграмма варинтов использования подсистемы

Из проанализированных методов тестирования и оценивания знаний знаний были выделены наиболее подходящие для использования в подсистеме тестирования знаний языков описания аппаратуры (таблица 3).

Таблица 3 — Методы тестирования знаний в спроектированной подсистеме

№	Тип	Подтип	Вид обратной связи
1	Тестирование с ответом в закрытой форме	Выбор одного ответа	Текстовое пояснение ошибки
		Выбор нескольких ответов	Информации о наличии ложноположительных (ложноотрицательных) ответов
2	Задание на написание исходного кода	Автоматизированное тестирование на проверяющей стороне	Информация о несоответствующих сигналах

На основе результатов проведенного анализа было заключено, что проектируемая подсистема должна выполнять следующие функции:

- изменение заданий модератором;
- отображение персональной статистики учащегося;
- обработка статистики решения заданий;
- формирование рейтингового списка учащихся;
- автоматизированная проверка тестов с закрытым ответом, кратким ответом и ответом в виде исходного кода;
- формирование информативной обратной связи в случае неверного решения задания учащимся;
- формирование временных диаграмм работы устройств.

На основе результатов проведенного анализа и сформулированных функциональных требований была разработана диаграмма вариантов использования подсистемы тестирования знаний языков описания аппаратуры, представленная на рисунке 8 [6].

1.4 Выводы

В исследовательской части проведен анализ функциональных возможностей существующих платформ обучения языкам программирования, получены и доработаны их функциональные модели, на их основе сформулированы функциональные требования и диаграмма вариантов использования подсистемы тестирования знаний языков описания аппаратуры.

Основными отличиями такой подсистемы от аналогов являются:

- реализация модуля автоматизированного тестирования на основе временных диаграмм, описываемых VCD-файлами;
- учет статистики прохождения заданий;
- информативная обратная связь об ошибках учащегося;
- формирование рейтинга учащихся.

Листинг 2 — Пример описания задания с множественным выбором

```
// условия задания с выбором нескольких вариантов ответа

{
  "caption": "Типы переменных в verilog",
  "answers": [
    "reg",
    "wire",
    "mem"
  ]
}

// ответ на задание с выбором нескольких вариантов ответа

{
  "correct_answers": [
    true,
    true,
    false
  ]
}
```

В случае задания на описание устройства с помощью языка Verilog, в поле LevelsData.question заносится код теста устройства на языке Verilog (т.н. «testbench», см. приложение Д), а в поле LevelsData.answer — описание временной диаграммы корректно описанного устройства в формате wavedrom (см. раздел «Генератор wavedrom-диаграмм»).

2.3 Проектирование микросервисов

2.3.1 Микросервис взаимодействия с БД

Для реализации CRUD-операций с данным, хранящимися в БД, был реализована микросервис взаимодействия с БД.

Логика работы с каждой из таблиц базы данных инкапсулирована в отдельный класс, каждый из таких классов работает с БД через класс соединения с БД, который в свою очередь использует драйвер СУБД MySQL (рисунок 13).

На рисунке 14 представлена диаграмма классов описываемого микросервиса.

Следующие классы реализуют взаимодействие с БД и воплощают в себе сущности предметной области:

- LevelsBrief;
- LevelsData;
- SolutionEffort;
- TypeRecord;
- User.

Класс MetaInfo содержит поля ObjType (сущность, надо которой выполняется операция) и Action (тип операции).

Классы с префиксом «Rf» (сокращение от «Request Frame») позволяют разобрать входные сообщения, разделив метаинформацию и данные о сущности предметной области.

Интерфейсы позволяют взаимодействовать с любым типом сущностей по одному и тому же алгоритму [8]:

- IReadable — интерфейс для чтения;
- ICreatable — интерфейс для записи;
- IUpdatable — интерфейс для обновления;
- IDeletable — интерфейс для удаления;
- IReadableAll — интерфейс для чтения всех записей;
- ICheckableSuccessful — интерфейс для проверки наличия корректных решений к заданию.

Использование типа interface для ResponseFrame.Data (данные ответного сообщения) так же позволяет записывать в это поле данные об объекте любого класса.

Примечание: любой класс в Golang является реализацией interface, однако не все отношения реализации показаны на диаграмме классов с целью ее упрощения.

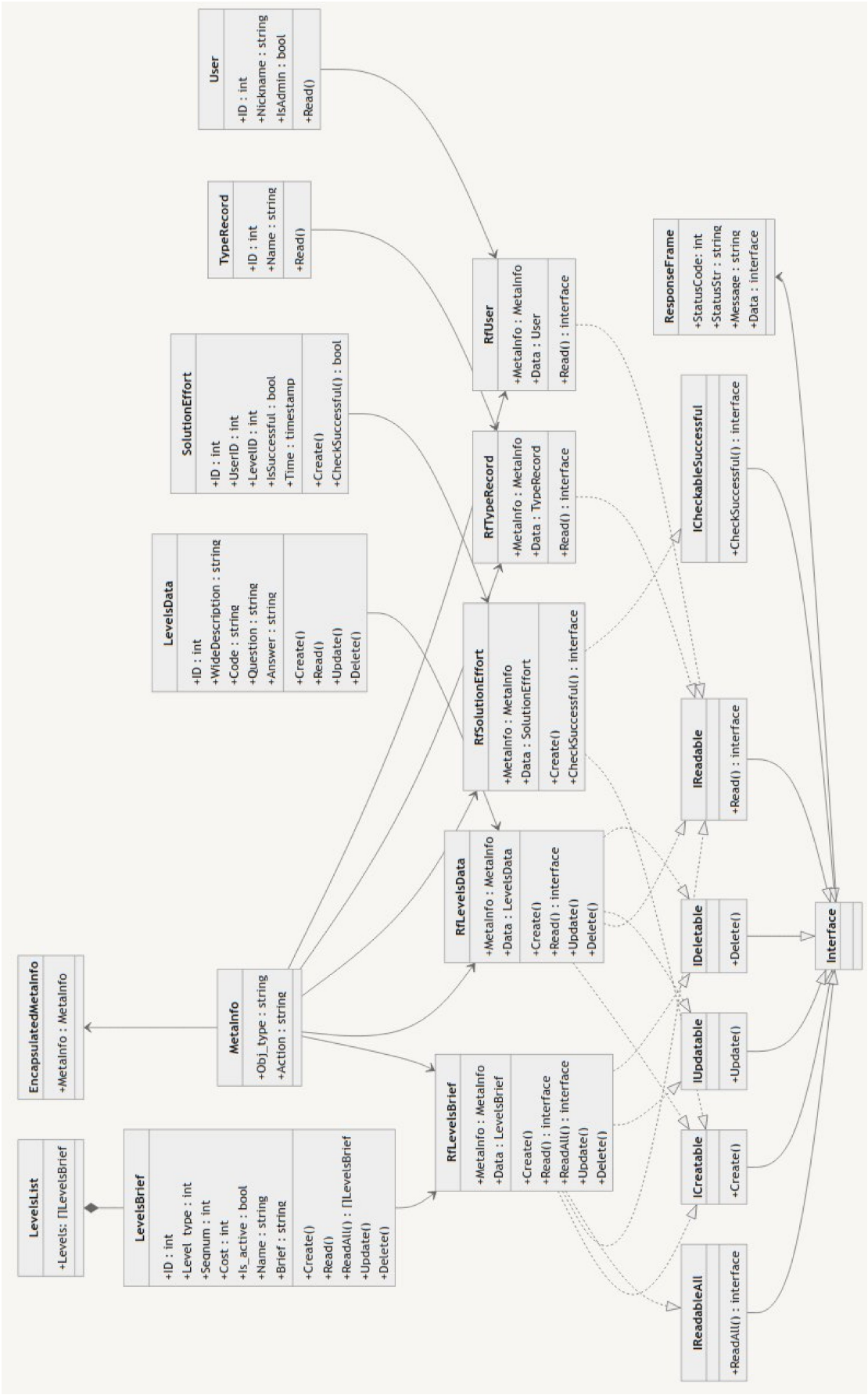

Рисунок 14 — диаграмма классов для работы с базой данных

Для универсализации алгоритма обработки решений используется полиморфизм (рисунок 20).

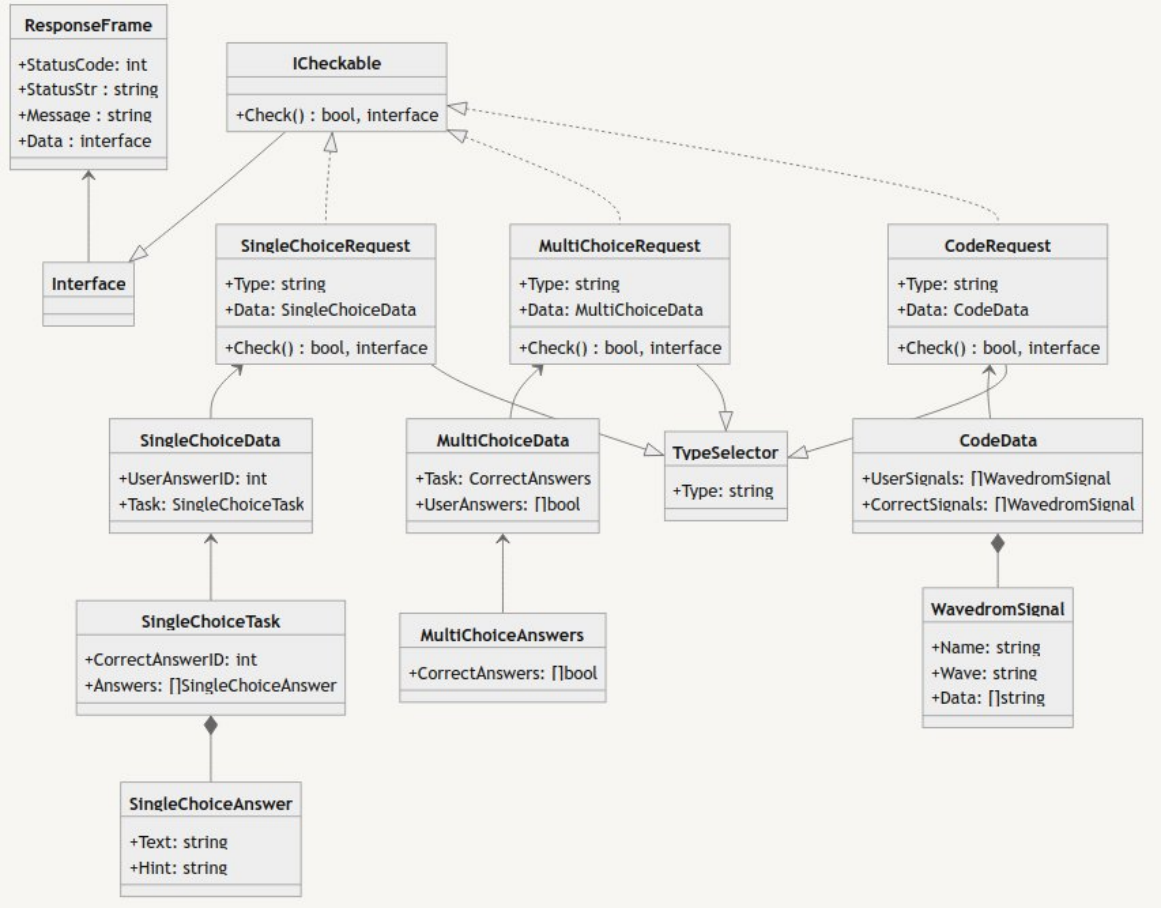

Рисунок 20 — Диаграмма классов анализатора решений

Исходный код этого микросервиса приведен в приложении Г.

2.4 Выводы

В рамках конструкторской части была спроектирована структура компонентов и спроектированы сами компоненты подсистемы тестирования знаний языков описания аппаратуры: подсистема взаимодействия с БД, синтезатор устройств, преобразователи формата временных диаграмм, анализатор решений и подсистема анализа статистики.

Спроектированные подсистема реализует функции и варианты использования, упомянутые в исследовательской части.

3.2 Разработка плана автономного тестирования

В качестве основного метода, используемого для разработки функциональных тестов, был использован метод эквивалентного разбиения.

Все задания, проверяемые анализатором решений, принадлежат к одному из следующих классов эквивалентности:

- тест с выбором одного ответа;
- тест с выбором множества ответа;
- задания на написания программы.

Для каждого из выделенных классов заданий можно выделить 3 класса решений:

- правильное решение задания;
- неправильное решение задания;
- решение, закодированное с нарушением формата.

На основе такого эквивалентного разбиения были составлены тесты, представленные в таблице 4.

Таблица 4 — Планируемые тесты анализатора

Номер теста	Название теста	Описание теста	Ожидаемый результат
1	single correct positive	Отправить на проверку одновариантный тест с ID ответа равным ID верного ответа	Признак is_correct = True
2	single correct negative	Отправить на проверку одновариантный тест с ID ответа не равным ID верного ответа	Признак is_correct = False

Продолжения таблицы 4

3	single error overflow	Отправить на проверку одновариантный тест с ID ответа больше максимального допустимого	Признак is_correct = False
4	multi correct positive	Отправить на проверку многовариантный тест с верными ответами	Признаки is_correct = True, false_positive = False, false_negative = False
5	multi correct false positive	Отправить на проверку многовариантный тест с ложноположительными ответами	Признаки is_correct = False, false_positive = True, false_negative = False
6	multi correct false negative	Отправить на проверку многовариантный тест с ложноотрицательными ответами	Признаки is_correct = False, false_positive = False, false_negative = True
7	multi error size mismatch	Отправить на проверку многовариантный тест с разной длиной массива ответов пользователя и эталонного ответа	Ошибка "answer size mismatch"
8	code correct positive	Отправить на проверку корректно выполненное задание на написание кода	Признак is_correct = True
9	code correct negative	Отправить на проверку некорректно выполненное задание на написание кода	Признак is_correct = False, возвращен список несовпадающих с эталоном сигналов

Микросервис взаимодействия БД работает с данными (метаданными) 4 классов (для каждой из сущностей):

- данные для создания экземпляра сущности;
- метаданные для чтения экземпляра сущности;
- данные для изменения экземпляра сущности;
- метаданные для удаления экземпляра сущности.

В каждом из этих классов могут содержаться следующие подклассы:

- корректные данные (метаданные);
- данные в некорректном формате (отсутствуют поля и т.п.);
- метаданные с несуществующим ID.

Так как классы эквивалентности одинаковы для всех сущностей, в таблице 5 приведены примеры тестов для сущности LevelsBrief. Тесты для других сущностей аналогичны.

Таблица 5 — Планируемые тесты микросервиса взаимодействия с БД для сущности LevelsBrief

Номер теста	Название теста	Описание теста	Ожидаемый результат
1	correct read level brief	Отправить запрос на считывание экземпляра сущности LevelsBrief с заданным ID из БД	Строка из БД с информацией о сущности LevelsBrief с заданным ID
2	correct create level brief	Отправить запрос на создание экземпляра сущности LevelsBrief, считать информацию о ней	Строка в БД с заданными значениями полей

Продолжение таблицы 5

3	correct update level brief	Отправить запрос на изменение экземпляра сущности LevelsBrief с заданным ID, считать информацию о ней	Строка в БД с новыми значениями полей
4	correct delete level brief	Отправить запрос на удаление экземпляра сущности LevelsBrief с заданным ID, считать информацию о ней	Поле is_archived = True
5	error create level invalid format	Отправить запрос на создание экземпляра сущности LevelsBrief, считать информацию о ней	Ошибка "invalid data format"
6	error read level invalid id	Отправить запрос на считывание экземпляра сущности LevelsBrief с несуществующим ID из БД	Ошибка "LevelsBrief entity with ID=<id> does not exist"

Для преобразователей временных диаграмм и синтезатора данные можно разделить на корректные и некорректные. Эти тесты приведены в таблицах 6-7.

Таблица 6 — Планируемые тесты микросервиса разбора временных диаграмм

Номер теста	Название теста	Описание теста	Ожидаемый результат
1	correct positive	Отправить запрос на преобразование временной диаграммы в формате VCD с корректными данными	JSON с описанием временной диаграммы в заданном формате

Продолжение таблицы 6

2	error in vcd	Отправить запрос на преобразование временной диаграммы в формате VCD с произвольными символами вместо корректных данных	Ошибка "vcd parsing error"

Таблица 7 — Планируемые тесты микросервис генерации временных диаграмм wavedrom

Номер теста	Название теста	Описание теста	Ожидаемый результат
1	correct positive	Отправить запрос на преобразование временной диаграммы в формате PyDigitalWaveTools	Временная диаграмма в формате wavedrom
2	error format	Отправить запрос на преобразование временной диаграммы с произвольными данными	Ошибка "invalid data format"

Для синтезатора в классе некорректных данных были выделены подклассы:

- данные с ошибкой в исходном коде устройства;
- данные с ошибкой в исходном коде тестов;
- данные без необходимой директивы "$dumpvars".

Соответствующие тесты приведены в таблице 8.

Таблица 8 — Планируемые тесты синтезатора

Номер теста	Название теста	Описание теста	Ожидаемый результат
1	correct positive	Отправить корректный исходный код описания устройства и тестов	Временная диаграмма в формате VCD
2	error in device	Отправить некорректный исходный код описания устройства и корректный код тестов	Ошибка "synthethis error"
3	error in testbench	Отправить корректный исходный код описания устройства и некорректный код тестов	Ошибка "simulation error"
4	error no dumpvars	Отправить корректный исходный код описания устройства и тестов, но без директивы dumpvars	Ошибка "testbench without $dumpvars"

Составленные автономные тесты позволяют обеспечить высокую степень покрытия функций компонентов разработанного ПО.

3.3 Разработка плана комплексного тестирования

Так как большинство базовых функций разработанной подсистемы тестирования знания было протестировано в режиме автономного тестирования, для проведения комплексного тестирования будет достаточно проверить пользовательские сценарии проверки задания, создания/изменения задания, возможность обращения к микросервисам анализа статистики и работы с БД.

Тесты для комплексного тестирования приведены в таблице 9.

Таблица 9 — Планируемые тесты основного микросервиса

Номер теста	Название теста	Описание теста	Ожидаемый результат
1	correct proxy crud	Запросить данные о пользователе с ID = 1 из БД	Данные о пользователе с ID = 1 из БД
2	correct proxy stats	Запросить статистику прохождения заданий на основе тестовых данных	Показатели статистики соответствуют предварительно рассчитанным
3	error no user in check	Запросить проверку задания для несуществующего пользователя	Ошибка "user does not exist"
4	error not admin in crud	Запросить изменение задания от имени пользователя, не являющегося администратором	Ошибка "user have no rights to modify levels"
5	correct check	Запросить проверку правильно выполненного задания	Данные о выполнении задания занесены в БД
6	error no level in check	Запросить проверку несуществующего задания	Ошибка "crud-microservice.levelsbrief error"
7	error create levelsdata	Запросить создание задания на программирование, указав некорректный исходный код описания устройства	Ошибка "device synthesis error"
8	correct create levelsdata	Запросить создание задания на программирование	Задание добавлено в БД

– полная обратная совместимость с unittest — возможность запуска тестов, написанных на нем;
– выполнение нескольких тестов параллельно;
– установочный код можно использовать повторно.

В дополнение к pytest была использована библиотека allure, формирующая интерактивные отчеты о прохождении тестов. Тесты в allure можно иерархически группировать и сопровождать логами и вложениями. Allure поддерживается не только для Python, но и для Java, JavaScript, Ruby, PHP , .Net и Scala.

Такой широкий набор поддерживаемых языков программирования делает allure (рисунок 22) знакомым многим разработчикам, тестировщикам и менеджерам, что упрощает поддержку тестов [15].

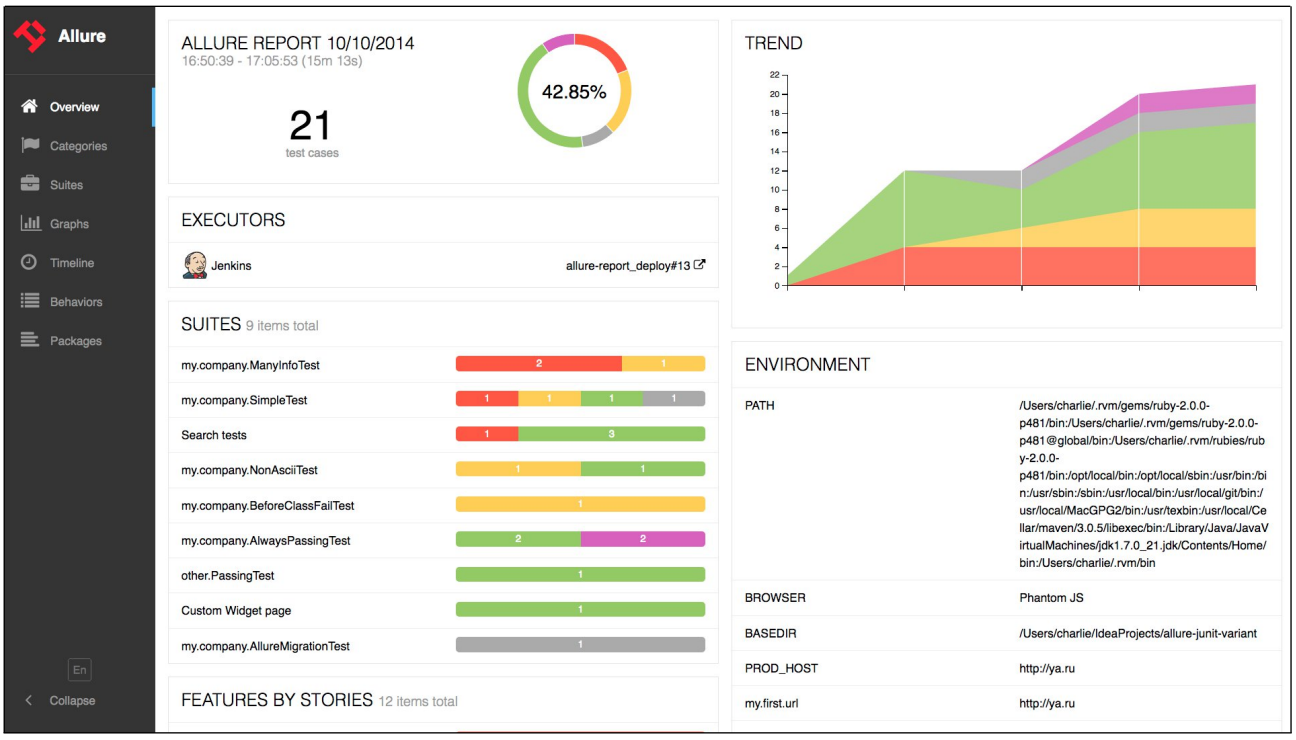

Рисунок 22 — Интерфейс allure

3.5 Реализация и проведение функциональных тестов

Для упрощения написания тестов и генерации отчетов был реализован вспомогательный модуль utils.py, отвечающий за отправку http-запросов к микросервисам, проверку http-ответов и их прикрепление к отчетам в allure. Программный код utils.py приведен в листинге 7.

визуализировать статитстику, допустимое среднее время ответа для запросов к БД было установлено равным 300 мс, а 95 перцентиль — 500 мс.

Результаты нагрузочного тестирования микросервиса разбора временных диаграмм приведены в таблице 10.

Таблица 10 — Результаты нагрузочного тестирования микросервиса разбора временных диаграмм

Статистика запросов							
Запросы	Ошиб.	Среднее (мс)	Мин. (мс)	Макс. (мс)	Сред. размер (байт)	RPS	Ошибки / с
18458	0	261	3	706	493	279.9	0.0
Статистика ответов							
50%ile (мс)	60%ile (мс)	70%ile (мс)	80%ile (мс)	90%ile (мс)	95%ile (мс)	99%ile (мс)	100%ile (мс)
260	300	350	400	450	470	520	710

Изменения частоты запросов, времени ответа и количества пользователей показаны на рисунках 27-28.

Рисунок 27 — Результаты нагрузочного тестирования микросервиса разбора временных диаграмм

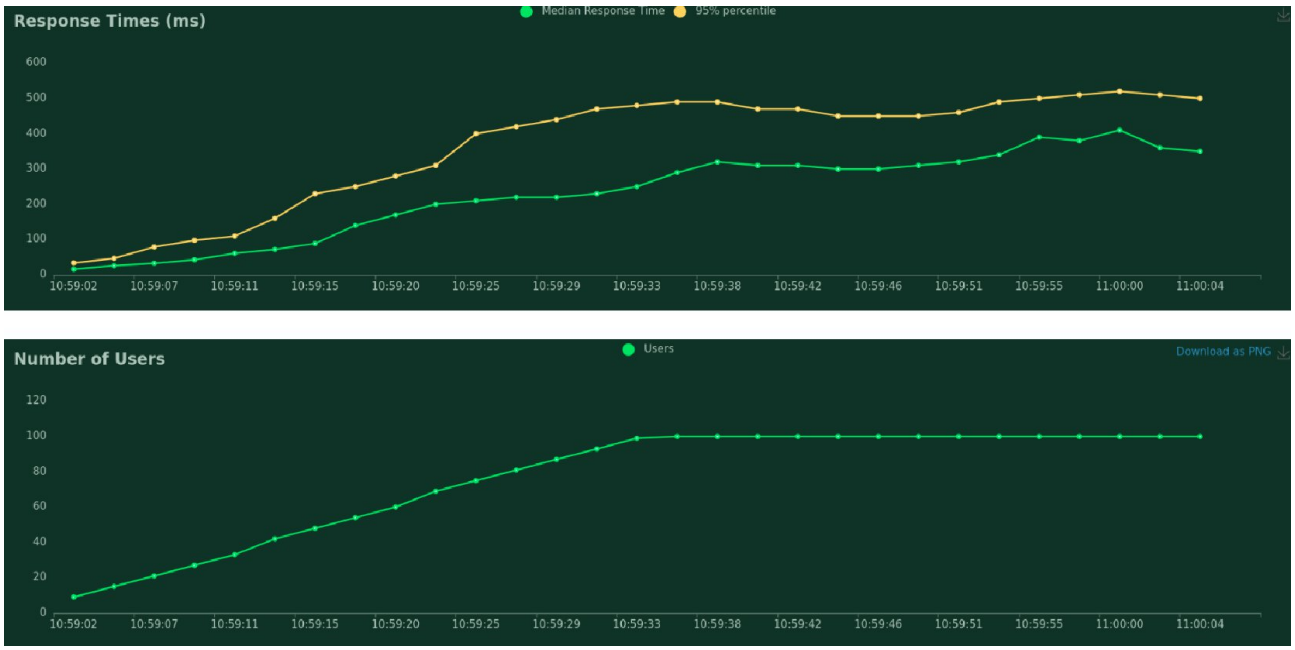

Рисунок 28 — Результаты нагрузочного тестирования микросервиса разбора временных диаграмм

Результаты нагрузочного тестирования синтезатора приведены в таблице 11.

Таблица 11 — Результаты нагрузочного тестирования синтезатора

Статистика запросов							
Запросы	Ошиб.	Среднее (мс)	Мин. (мс)	Макс. (мс)	Сред. размер (байт)	RPS	Ошибки / с
12936	0	525	13	1339	3050	152.8	0.0
Статистика ответов							
50%ile (мс)	60%ile (мс)	70%ile (мс)	80%ile (мс)	90%ile (мс)	95%ile (мс)	99%ile (мс)	100%ile (мс)
580	610	640	680	770	870	950	1300

Изменения частоты запросов, времени ответа и количества пользователей показаны на рисунке 29.

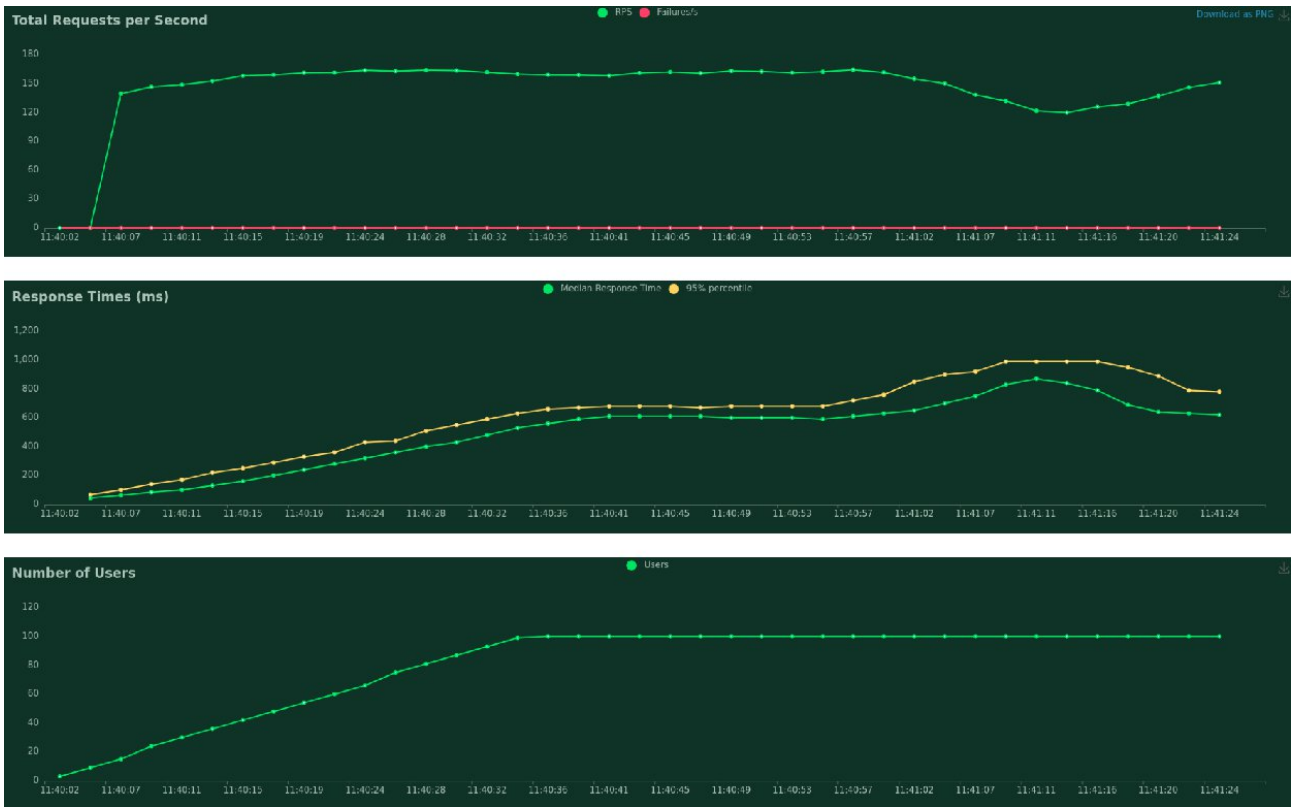

Рисунок 29 — Результаты нагрузочного тестирования синтезатора

Результаты нагрузочного тестирования обращения к БД через слой бизнес-логики приведены в таблице 12.

Таблица 12 — Результаты нагрузочного тестирования обращения к БД через слой бизнес-логики

Статистика запросов								
Марш.	Запросы	Ошибки	Среднее (мс)	Мин. (мс)	Макс. (мс)	Сред. размер (байт)	RPS	Ошибки / с
/levels	6743	0	85	3	676	506	111.8	0.0
/stats	20486	0	161	2	1151	331	339.5	0.0
Итого	27229	0	142	2	1151	374	451.3	0.0

Продолжение таблицы 12

Статистика ответов								
Марш.	50%ile (мс)	60%ile (мс)	70%ile (мс)	80%ile (мс)	90%ile (мс)	95%ile (мс)	99%ile (мс)	100%ile (мс)
/levels	78	94	110	130	160	180	220	680
/stats	110	140	190	270	370	460	630	1200
Итого	100	130	160	210	330	430	600	1200

Изменения частоты запросов, времени ответа и количества пользователей показаны на рисунке 30.

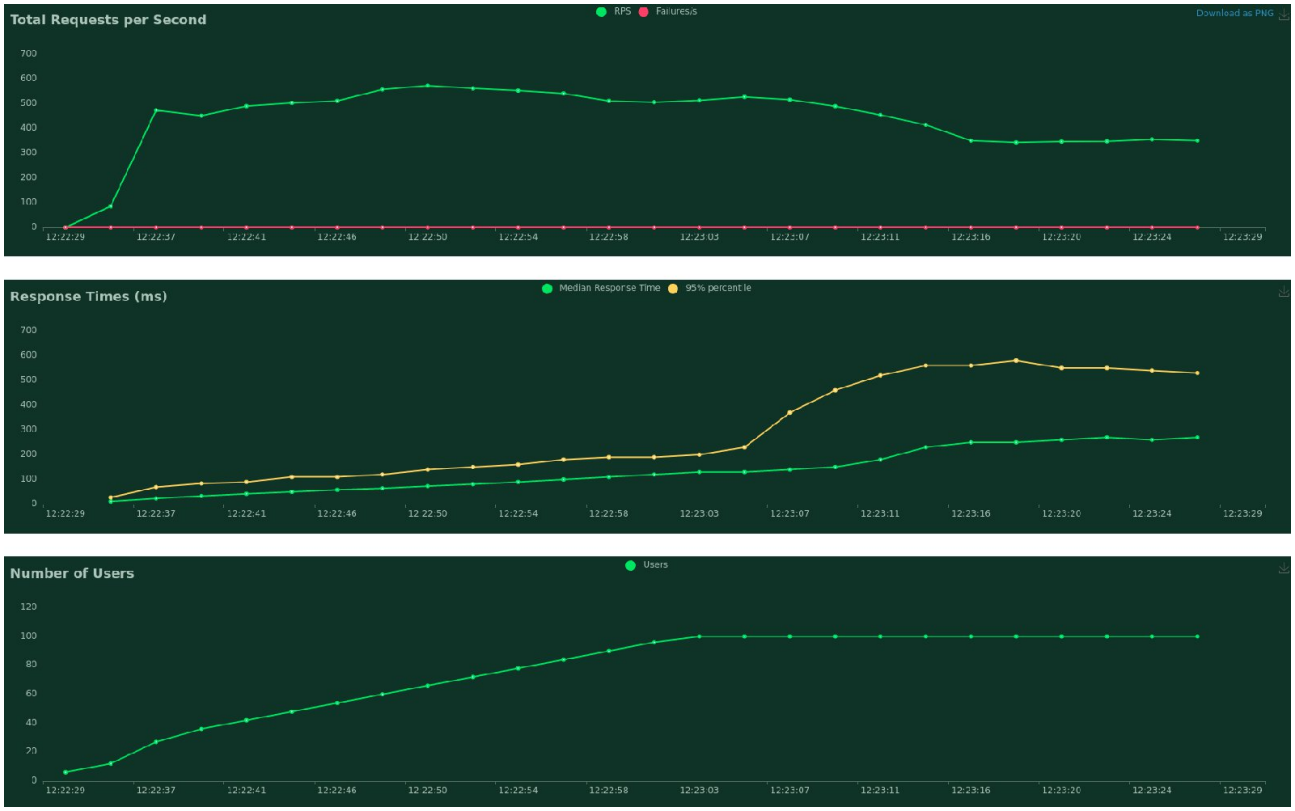

Рисунок 30 — Результаты нагрузочного тестирования обращения к БД через слой бизнес-логики

Тестируемые компоненты в ходе тестирования работали корректно, полученные задержки находятся в допустимых пределах.

Приложение В

Графические материалы

Листов 7

Перечень графичекого материала:

1) Схема структурная ВКРБ;
2) Диаграмма вариантов использования подсистемы тестирования знаний;
3) Схема структурная информационной системы;
4) Даталогичесчкая модель БД. Диаграмма компоновки микросервиса взаимодействия с БД;
5) Диаграмма классов для взаимодействия с базой данных;
6) Диаграмма компоновки микросервиса синтеза устройств. Диаграмма классов микросервиса анализа решений;
7) Результаты тестирования подсистемы.