Оглавление

Глоссарий

- комплекс
- сус
- сбн
- су
-
- задача
- пользователь
- вычислительный узел
-
- х?
-
-

Введение

1. Аналитический раздел

1.1. Введение

В данном разделе выполняется анализ предметной области. Результаты анализа представляются в виде диаграм прецедентов и деятельности.

1.2. Возможные прецеденты

Комплекс при его работе предоставляет пользователю следующие варианты использования:

- регистрация пользователя;
- авторизация пользователя;
- постановка задачи на исполнение;
- просмотр статуса задачи;
- отмена задачи.

Диаграмма этих и дополнительных служебных прецедентов приведена на рис. 1.

С учётом требований к разделению внутреннего функционала комплекса, диаграмма прецедентов на рис. 1 расщепляется на набор диаграмм, соответствующих каждой из выделенных подсистем. Соответствующие диаграммы приведены на рисунках 2,3,4.

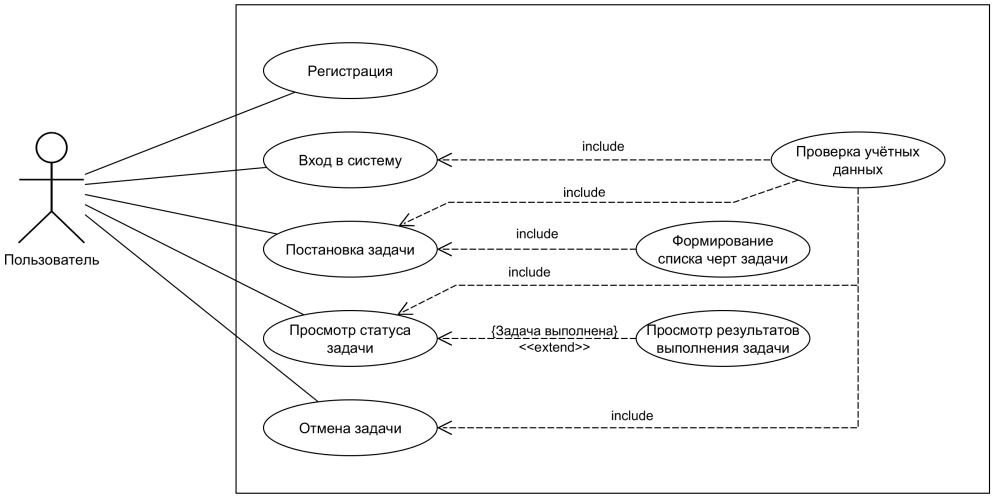

Рис. 1: Диаграмма прецедентов всего комплекса в целом

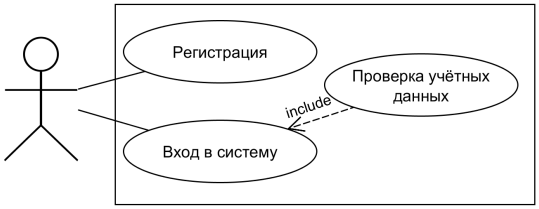

Рис. 2: Диаграмма прецедентов СУС

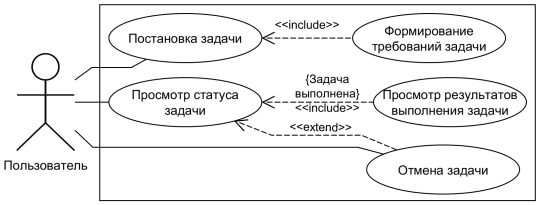

Рис. 3: Диаграмма прецедентов СУ

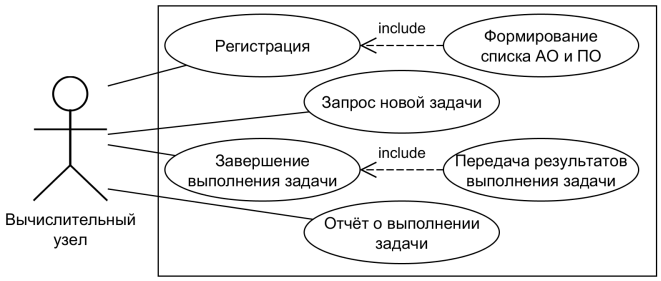

Рис. 4: Диаграмма прецедентов СБН

1.3. Осуществляемая деятельность

Прецеденты, описанные в предыдущем пункте, отвечают определённой деятельности. Диаграмма деятельности на рис. 5 описывает полный процесс взаимодействия пользователя с комплексом.

С учётом требований к разделению внутреннего функционала комплекса, диаграмма деятельности на рис. 5 расщепляется на набор диаграмм, соответствующих определённым подсистемам из выделенных.

Диаграммы действий прецедентов подсистемы управления сессией “регистрация” и “вход в систему” приведены на рисунках 6 и 7 соответственно.

Диаграммы действий прецедентов системы балансировки нагрузки “регистрация”, “запрос новой задачи” и “завершение выполнения задачи” приведены на рисунках 8, 9 и 10 соответственно.

Диаграммы действий прецедентов системы управления “постановка задачи” и “просмотр статсуа задачи” приведены на рисунках 11 и 12 соответственно.

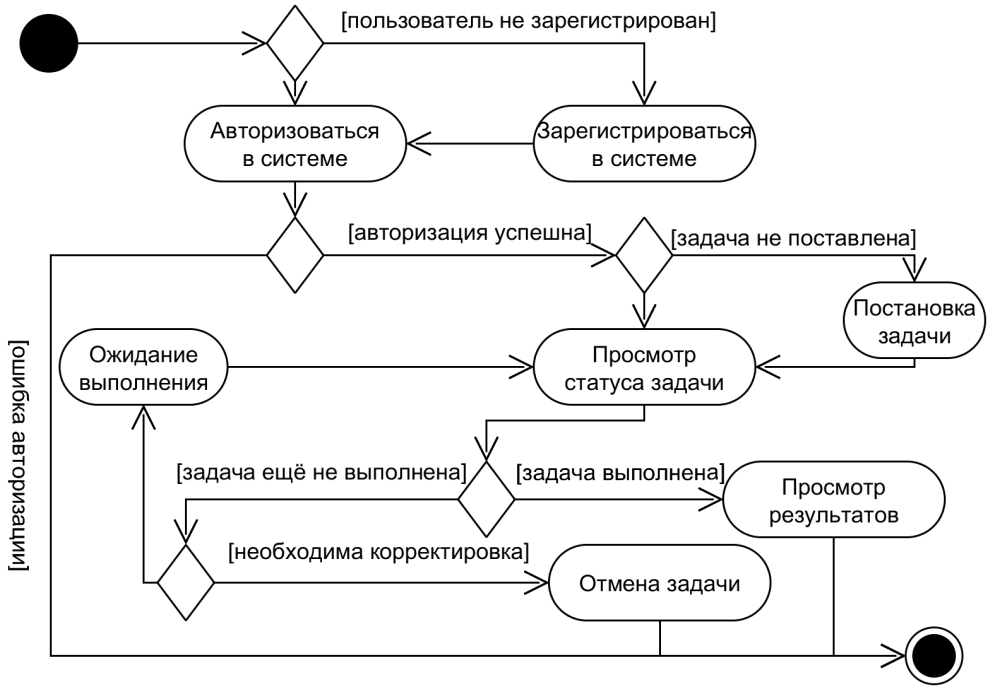

Рис. 5: Диаграмма действий прецедента “общая деятельность” для системы в целом

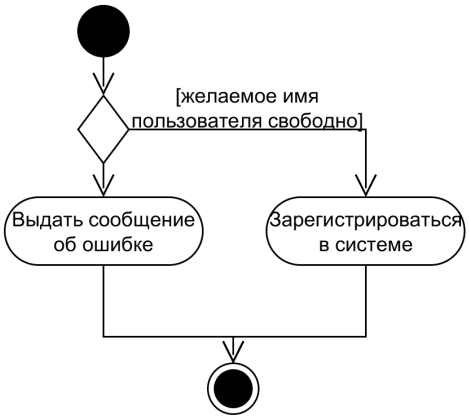

Рис. 6: Диаграмма действий прецедента “регистрация” СУС

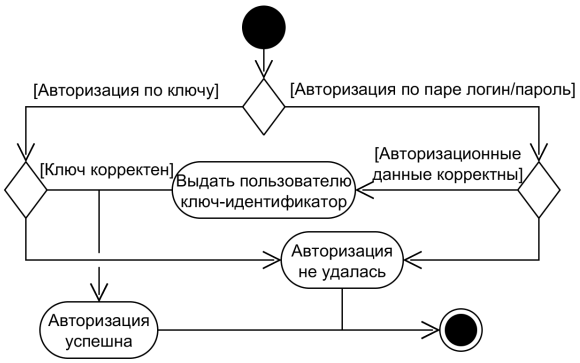

Рис. 7: Диаграмма действий прецедента “вход в систему” СУС

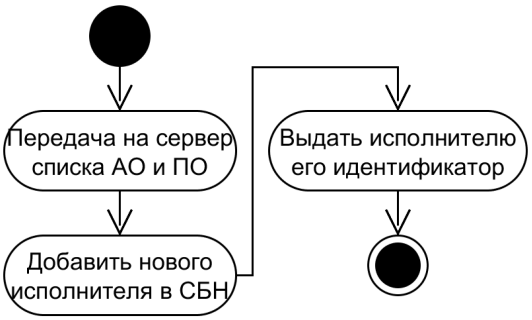

Рис. 8: Диаграмма действий прецедента “регистрация” СБН

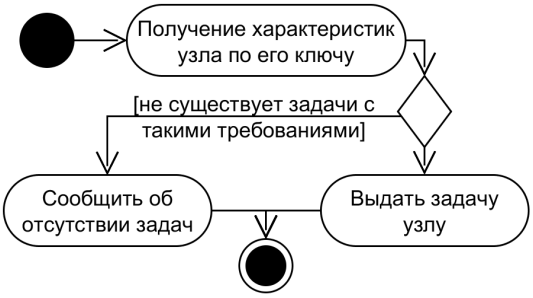

Рис. 9: Диаграмма действий прецедента “запрос новой задачи” СБН

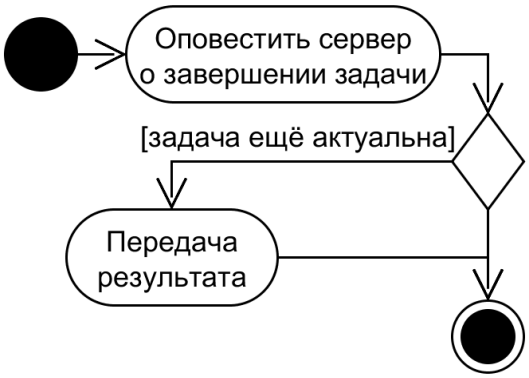

Рис. 10: Диаграмма действий прецедента “завершение выполнения задачи” СБН

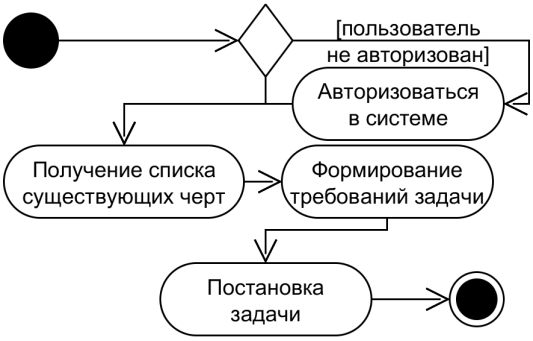

Рис. 11: Диаграмма действий прецедента “постановка задачи” СУ

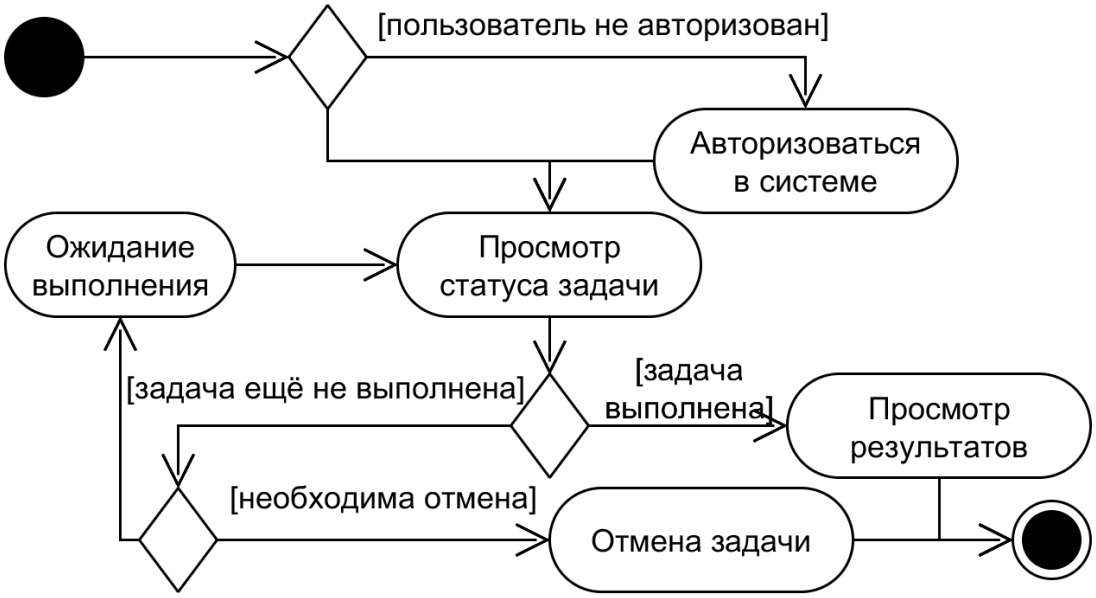

Рис. 12: Диаграмма действий прецедента “просмотр статсуа задачи” СУ

1.4. Вывод

В данном разделе были приведены диаграммы, описывающие функционал основных узлов системы. Данный анализ в дальнейшем используется для более строгой формализации функционала подсистем.

2. Конструкторский раздел

2.1. Введение

В данном разделе приводятся результаты проектирования системы. С применением UML-диаграмм описывается общая структура комплекса и требуемый функционал отдельных узлов системы.

2.2. Общая структура системы

Для того, чтобы удовлетворить требованиям по предоставлению механизма деградации функциональности, а также для упрощения процесса разработки, комплекс должен быть разделен на отдельные слабосвязанные элементы.

Различные подсистемы комплекса имеют некую модель поведения. Поведение подсистемы описывается её активной и пассивной частью. Активная часть соответствует действиям, которые подсистема выполняет разово либо с некоторой периодичностью, в автоматическом режиме. Пассивная часть соответствует API подсистемы. Взаимосвязи между различными компонентами системы приведены на диаграмме компонентов на рис. 13. Физическое размещение компонент по отдельным узлам проиллюстрировано на диаграмме развёртывания на рис. 14.

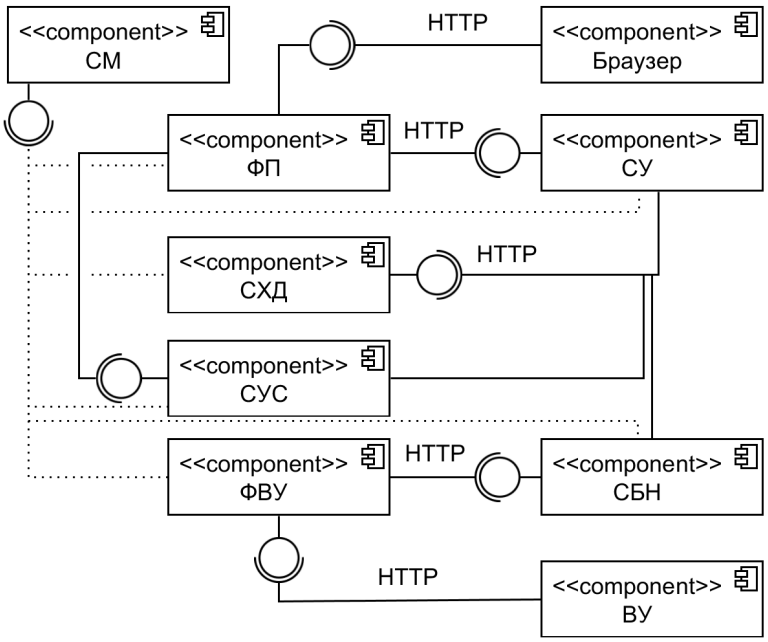

Рис. 13: Диаграмма компонент комплекса

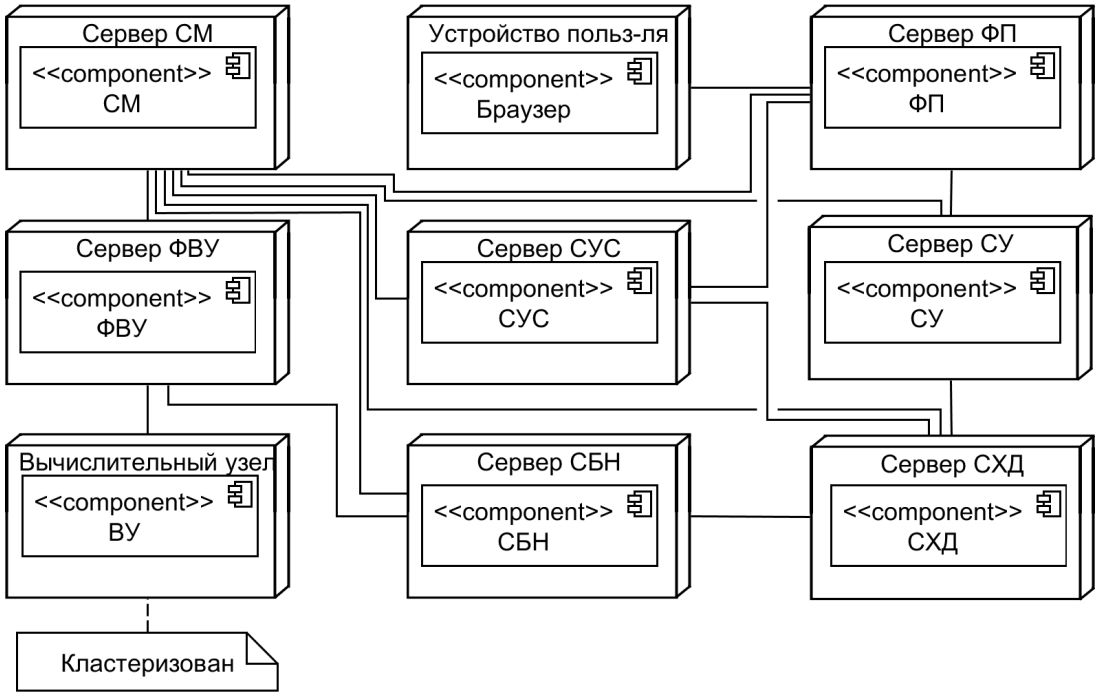

Рис. 14: Диаграмма развёртывания комплекса

2.3. Система мониторинга

Задача данной подсистемы – отслеживание топологии сети. Все узлы комплекса должны оповещать СМ о своём статусе работы, и любой узел может получить от комплекса список активных в данный момент узлов. Данная система является полностью пассивной.

Невозможность любой другой подсистемы связаться с системой мониторинга рассматривается как ошибка сети, нарушающая нормальное функционирование комплекса.

Пассивная часть

Исходя из требований к СМ и с учётом REST-методик, она должна предоставлять следующее API:

- **Ресурс:** `/services`

  Метод: GET

  Результат: список активных сервисов

- **Ресурс:** `/services/type`

  Метод: GET

  Результат: список активных сервисов такого типа

- **Ресурс:** `/services/type`

  Метод: POST

  Параметры: port, state?

  Результат: сообщение об успешной регистрации сервиса и распознанный адрес сервиса

  Ошибки: отсутствует параметр 'port': HTTP 422

- **Ресурс:** `/services/type/address`

  Метод: GET

  Результат: статусное сообщение выбранного сервиса, аннотированное временем создания

  Ошибки: сервис не найден: HTTP 404

- **Ресурс:** `/services/type/address`

  Метод: PUT

  Параметры: state?

```
{
  ”type1”: {
    ”ipaddr1:port1”: {
      ”state”: ”message1”,     // arbitrary  string
      ”lastbeat”: ”datetime1” // ex. ”2015−05−25 01:46:35.857670”
    },
    ”ipaddr2:port2”: {
      ”state”: ”message2”,
      ”lastbeat”: ”datetime2”
    }
  },
  ”type2”: {
    ”ipaddr3:port3”: {
      ”state”: ”message3”,
      ”lastbeat”: ”datetime3”
    }
  }
}
```

Листинг 1: Пример JSON-представления коллекции верхнего уровня сервиса мониторинга

Результат: сообщение об успешном обновлении статусного сообщения

Коллекция верхнего уровня `/services` имеет словарь JSON-формата, приведённого на примере в листинге 1. Запросы к коллекциям более глубокого уровня `/services/type` и `/services/type/address` отображаются в подсловари root[type] и root[type][address] словаря верхнего уровня, соответственно.

2.4. Фронтэнд пользователей

Задача данной подсистемы – проверки безопасности и перенаправление запросов от пользователей к системе управления, а также отрисовка веб-интерфейса.

Активная часть

- В ходе конфигурирования данной системы необходимо в ручном порядке указать адрес системы мониторинга.
- ФП должен зарегистрироваться на СМ и оповещать её о своём состоянии с некоторой периодичностью.

- В ходе работы ФП должен получать со стороны СМ информацию о текущем адресе СУ и СУС.

Пассивная часть

Исходя из требований к ФП, он должен предоставлять следующий функционал:

- регистрация пользователя
- авторизация пользователя
- просмотр списка существующих в системе черт
- постановка задачи с передачей в систему описания задачи, содержащего:

  файл со списком черт задачи

  все необходимые для запуска задачи файлы

  в зависимости от платформы, либо файл run.bat, либо файл run.sh, осуществляющий запуск программы

  список выходных файлов задачи в файле output.txt
- просмотр статусов поставленных задач
- для выполненных задач – просмотр результатов выполнения
- удаление задач

На файлы, составляющие описание задачи, накладываются следующие ограничения:

- Все файлы, кроме файла traits.txt, должны содержаться в zip-архиве.
- Файл traits.txt должен содержать набор строк в кодировке UTF-8. В каждой строке должна быть записана очередная черта задачи. Черта задачи состоит из имени черты и версии черты, разделёнными символом табуляции Файл может заканчиваться пустой строкой. Пример корректного файла дан в листинге 2.
- Файл output.txt должен содержать набор строк в кодировке UTF-8. В каждой строке должен быть записан путь к одному из выходных файлов. Началом пути считается корень архива. Файл может заканчиваться пустой строкой. Пример корректного файла дан в листинге 3.

```
root_folder_file.txt
directory/subdirectory_file.extension

```

Листинг 2: Пример корректного файла со списком черт задачи

```
root_folder_file.txt
directory/subdirectory_file.extension

```

Листинг 3: Пример корректного файла со списком выходных файлов

- Файл run.bat / run.sh должен содержать инструкции по запуску задачи и корректно исполняться на вычислительном узле, удовлетворяющем требованиям, описанных чертами задачи. Именно этот файл будет запущен вычислительным узлом для начала расчётов. По завершении выполнения задачи она должна закрыть все созданные окна и уничтожить все порождённые процессы.

2.5. Фронтэнд вычислительных узлов

Задача данной подсистемы – перенаправление запросов от вычислительных узлов на балансировщик нагрузки.

Активная часть

- В ходе конфигурирования данной системы необходимо в ручном порядке указать адрес системы мониторинга.
- ФВУ должен зарегистрироваться на СМ и оповещать её о своём состоянии с некоторой периодичностью.
- В ходе работы ФВУ должен получать со стороны СМ информацию о текущем адресе СБН.

Пассивная часть

Исходя из требований к ФВУ и с учётом REST-методик, он должен предоставлять следующее API:

- **Ресурс:** `/nodes`

Метод: POST

Параметры: список черт вычислительного узла

Результат: сообщение об успешной регистрации узла и назначенный идентификатор

- **Ресурс:** `/nodes/nodeid`

  Метод: PUT

  Параметры: состояние расчёта

  Результат: сообщение об успешном обновлении статуса

- **Ресурс:** `/tasks/newtask`

  Метод: GET

  Параметры: идентификатор вычислительного узла

  Результат: пакет данных, описывающих задачу

  Ошибки: подходящих задач нет: HTTP 404; идентификатор не распознан либо отсутствует: HTTP 422

- **Ресурс:** `/tasks/taskid`

  Метод: POST

  Параметры: идентификатор вычислительного узла, результат выполнения задачи

  Результат: сообщение об успешном приёме результата

  Ошибки: ошибка идентификатора узла, формата задачи либо идентификатора задачи: HTTP 422

Методы API POST `/nodes`, PUT `/nodes/nodeid`, GET `/tasks/newtask` и POST `/tasks/taskid` соответствуют прецедентам “регистрация”, “запрос новой задачи”, “отчёт о выполнении задачи” и “завершение выполнения задачи” соответственно.

Диаграммы последовательности действий при выполнении прецедентов “регистрация”, “запрос новой задачи” и “завершение выполнения задачи” приведены на рис. 15, 16 и 17.

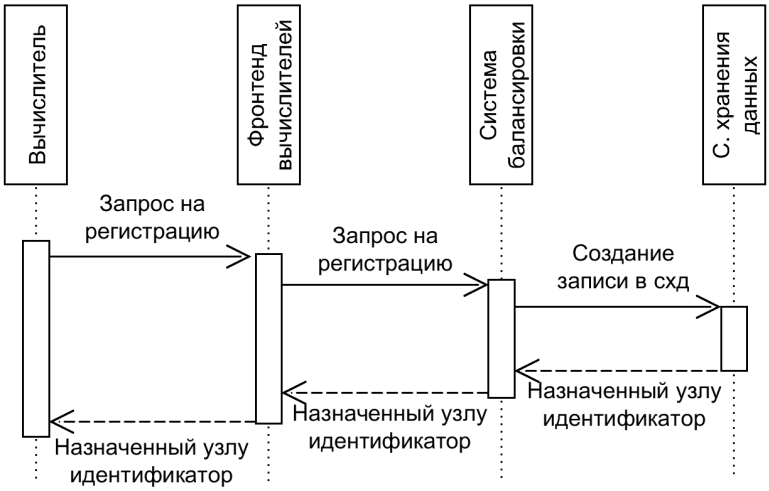

Рис. 15: Диаграмма последовательности действий прецедента “регистрация” ФВУ

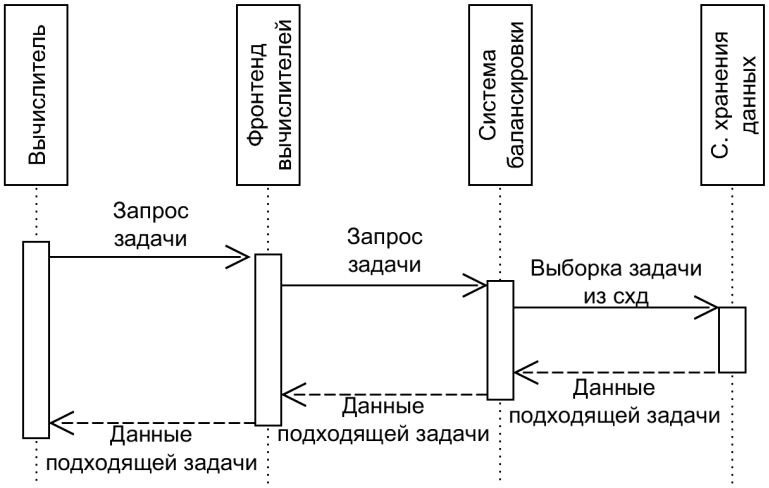

Рис. 16: Диаграмма последовательности действий прецедента “запрос новой задачи” ФВУ

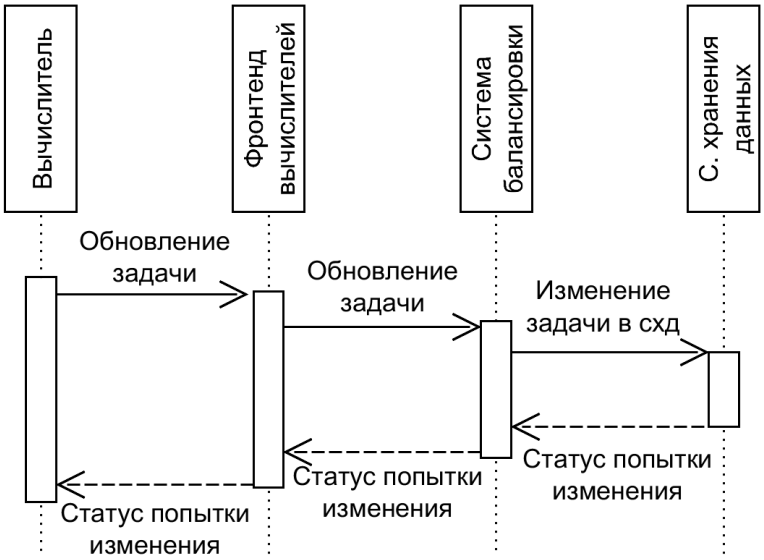

Рис. 17: Диаграмма последовательности действий прецедента “завершение выполнения задачи” ФВУ

2.6. Система управления сессией

Задача данной подсистемы – регистрация, авторизация и аутентификация пользователей в сети.

Активная часть

- В ходе конфигурирования данной системы необходимо в ручном порядке указать адрес системы мониторинга.
- СУС должна зарегистрироваться на СМ и оповещать её о своём состоянии с некоторой периодичностью.

Пассивная часть

Исходя из требований к СУС и с учётом REST-методик, она должна предоставлять следующее API:

- **Ресурс:** `/users`

  Метод: POST

  Параметры: желаемая пара логин / пароль (возможно, хешированный)

Результат: сообщение об успешной регистрации пользователя

Ошибки: пользователь с таким именем уже зарегистрирован: HTTP 403

- **Ресурс:** `/users/username`

  Метод: GET

  Параметры: пароль (возможно, хешированный)

  Результат: сгенерированный ключ доступа

  Ошибки: некорректная пара логин / пароль: HTTP 403

- **Ресурс:** `/validate`

  Метод: GET

  Параметры: ключ доступа

  Результат: сообщение об успешной проверке ключа

  Ошибки: некорректный ключ: HTTP 401

2.7. Система управления

Задача данной системы – предоставление API, позволяющего интерфейсной части (фронтэнду вычислительных узлов) осуществлять взаимодействие пользователя с комплексом.

Активная часть

- В ходе конфигурирования данной системы необходимо в ручном порядке указать адрес системы мониторинга.
- СУ должна зарегистрироваться на СМ и оповещать её о своём состоянии с некоторой периодичностью.
- В ходе работы СУ должна получать со стороны СМ информацию о текущем адресе СХД.

Пассивная часть

Исходя из требований к СУ и с учётом REST-методик, она должна предоставлять следующее API:

- **Ресурс:** `/foo`

  Метод: BAR

Параметры: lorem

Результат: ipsum

Ошибки: dolor

2.8. Система хранения данных

Задача данной системы – хранение данных о задачах, вычислительных узлах и их чертах, а также предоставление API по доступу к этим данным.

Связи между разными типами хранимых данных предоставлены в виде ER-диаграммы на рис. 18.

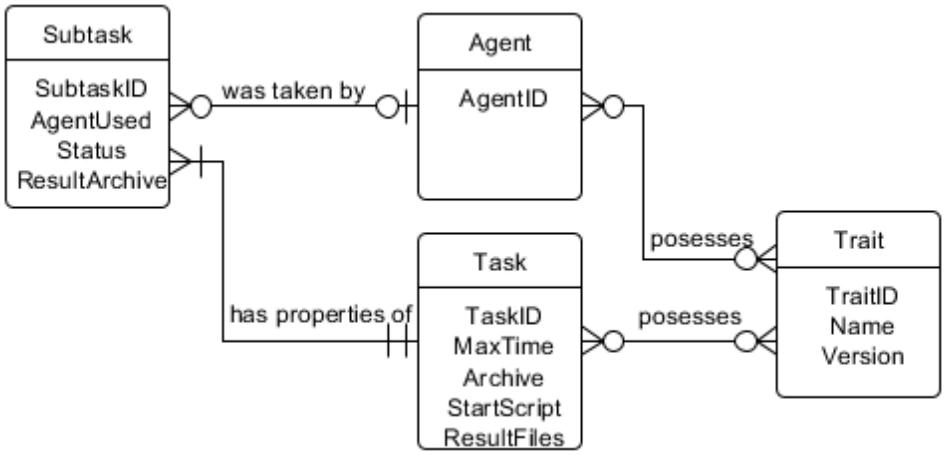

Рис. 18: ER-диаграмма сущностей, хранимых в СХД

Активная часть

- В ходе конфигурирования данной системы необходимо в ручном порядке указать адрес системы мониторинга.
- СХД должна зарегистрироваться на СМ и оповещать её о своём состоянии с некоторой периодичностью.

Пассивная часть

Исходя из требований к СХД и с учётом REST-методик, она должна предоставлять следующее API:
Условные обозначения:

- table – имя таблицы в БД

- PKC – поле, являющееся (целочисленным) первичным ключом
- GFSbAI – get_free_subtask_by_agent_id
- object – {field: value, ...}, JSON-представление табличного объекта
- object <table> – JSON-предтавление объекта определенной таблицы <table>
- value <name> – значение, передаваемое любым возможным способом
- arrfilter object – {field: list value, ...}
- filter object – object, содержащий не более чем полный набор полей табличного объекта
- filter put object – {field:value, ..., “changes”: filter object}
- list object – список объектов
- int – целое число

Адрес	Метод	Ввод	Код	Вывод
/table	POST	object	200	empty
			400	error: Incorrect / insufficient input
			500	error: Postgres error or “entry already exists”
	GET	–	200	{“result”: list object}
			404	error: Not found
/table/filter	GET	filter object	200	{“result”: list object}
			400	error: Incorrect fields/values specified
	PUT	filter put object	200	{“count”: int}
			400	error: Incorrect fields/values specified
	DELETE	filter object	200	{“count”: int}
			400	error: Incorrect fields/values specified

/table/arrayfilter	GET	arrfilter object	200	{“result”: list object}
			400	error: Incorrect fields/values specified
/table/PKC/value	GET	object	200	object
			400	error: Incorrect fields/values specified
			404	error: Not found
	PUT	object	200	object
			400	error: Incorrect fields/values specified
			404	error: Not found
	DELETE	object	200	object
			400	error: Incorrect fields/values specified
			404	error: Not found
/custom/GFSbAI	GET	value agent_id	200	object subtask
			400	error: Incorrect value specified

Кроме указанных кодов ошибок, также все бекенды могут вернуть в ответ:

- 408 – Таймаут попытки доступа к некоторому шарду;
- 456 – Многочисленные(различные) ошибки полученные от шардов при выполнении запроса. Эта ошибка является следствием нарушения согласованности данных.

2.9. Система балансировки нагрузки

— собственно отвечает за координацию задач. имеет всё апи фронтенда вычислительных узлов (который просто редиректит запросы к ней), плюс некоторое апи по которому её опрашивают другие узлы комплекса.

Активная часть

- В ходе конфигурирования данной системы необходимо в ручном порядке указать адрес системы мониторинга.

- СБН должна зарегистрироваться на СМ и оповещать её о своём состоянии с некоторой периодичностью.
- В ходе работы СБН должна получать со стороны СМ информацию о текущем адресе СХД.

Пассивная часть

Исходя из требований к СБН и с учётом REST-методик, она должна предоставлять API, идентичное предоставляемому фронтэндом вычислительных узлов.

2.10. Система вычисления

Данная система представлена набором вычислительных узлов с установленным на них специальным ПО, осуществляющем взаимодействие с остальными сервисами системы и управление ходом выполнения задачи.

Активная часть

В ходе конфигурирования данной системы необходимо в ручном порядке указать адрес фронтэнда вычислительных узлов.

ПО, обеспечивающее функционирование системы, должно удовлетворять следующим требованиям:

- До подключения к серверу балансировки приложение должно предоставлять возможность формирования списка черт, характеризующих АО и ПО вычислительного узла
- После подключения к балансировщику (через фронтэнд вычислительных узлов), с определённой периодичностью вычислительный узел должен опрашивать комплекс на предмет наличия доступных задач
- По получении задачи, вычислительный узел должен с определённой периодичностью оповещать балансировщик о ходе выполнения задачи
- По завершении выполнения задачи, вычислительный узел должен передать балансировщику сведения о результате выполнения задачи

Диаграмма состояний ПО вычислительного узла, иллюстрирующая приведённые выше соображения, приведена на рис. 19.

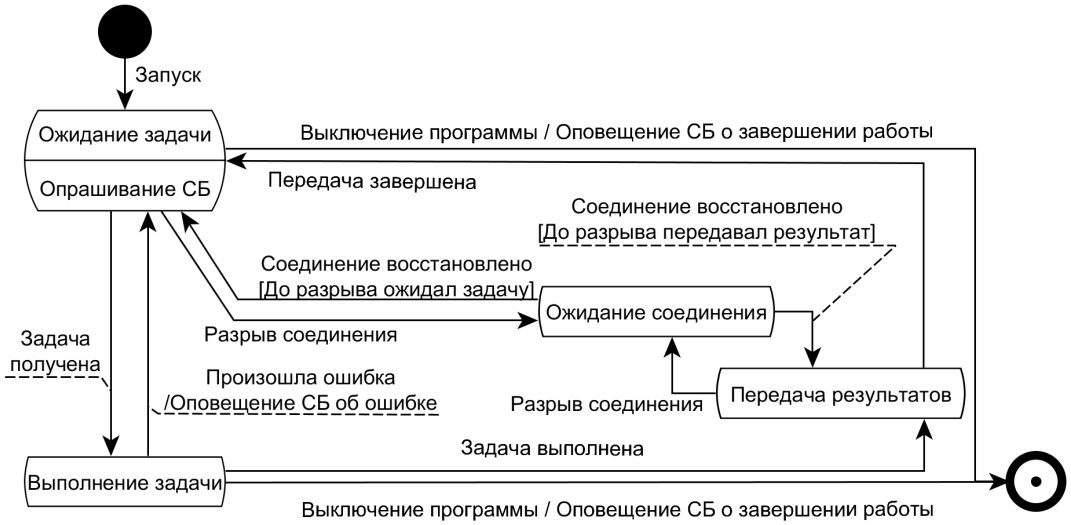

Рис. 19: Диаграмма состояний ВУ

2.11. Вывод

3. Технологический раздел

3.1. Введение

В данном разделе производится выбор языка программирования и сопутствующих программных средств. Описываются основные моменты программной реализации и описывается методика тестирования.

3.2. Выбор языка программирования

3.3. Выбор программных средств

перечень средств: python flask jsonpickle

3.4. Система мониторинга

Реализация

Система была реализована с помощью python-фреймворка flask. Для сериализации данных в json-формат и обратно была использована библиотека jsonpickle. Исходный код системы приведён в приложении [необходимо разобраться с автонумерацией приложений].

Тестирование

В ходе юнит-тестирования проверялись следующие сценарии:

- доступ к отсутствующему элементу через ресурсы `/services`, `/services/type` и `/services/type/address`;
- попытка создания записи по ресурсу `/services/test` без указания порта;
- попытка создания записи по ресурсу `/services/test`;
- выборка всех активных сервисов по ресурсу `/services`;
- выборка активных сервисов роли test по ресурсу `/services/test`;
- изменение статусного сообщения сервиса по ресурсу `/services/test/address`;
- проверка удаления сервиса из списка активных после определённого времени неактивности.

Тест был выполнен на языке Python. Выход теста приведён в листинге 4. Соответствующие сообщения сервера приведены в листинге 5.

```
...\grid_calc\src\python\beacon_backend\tests>py beacon_backend_ut.py
#1: Running empty run test... [PASSED]
#2: Running incorrect POST placing test... [PASSED]
#3: Running correct POST placing test... [PASSED]
#4: Running root request test... [PASSED]
#5: Running role request test... [PASSED]
#6: Running data modification test... [PASSED]
#7: Running specific access test... [PASSED]
#8: Running expiration test... [PASSED]
All tests passed.
```

Листинг 4: Выход юнит-теста системы мониторинга

```
...\grid_calc\src\python\beacon_backend>py beacon_backend.py
Port number defaulted to 666
 * Running on http://0.0.0.0:666/ (Press CTRL+C to quit)
127.0.0.1 - - [25/May/2015 00:28:36] "GET /services HTTP/1.1" 200 -
127.0.0.1 - - [25/May/2015 00:28:36] "GET /services/test HTTP/1.1" 200 -
127.0.0.1 - - [25/May/2015 00:28:36] "GET /services/test/127.0.0.1:345 HTTP/1.1"
    404 -
127.0.0.1 - - [25/May/2015 00:28:36] "POST /services/test HTTP/1.1" 422 -
127.0.0.1 - - [25/May/2015 00:28:36] "POST /services/test HTTP/1.1" 200 -
127.0.0.1 - - [25/May/2015 00:28:36] "GET /services HTTP/1.1" 200 -
127.0.0.1 - - [25/May/2015 00:28:36] "GET /services/test HTTP/1.1" 200 -
127.0.0.1 - - [25/May/2015 00:28:36] "PUT /services/test/127.0.0.1:345 HTTP/1.1"
    200 -
127.0.0.1 - - [25/May/2015 00:28:36] "GET /services/test/127.0.0.1:345 HTTP/1.1"
    200 -
127.0.0.1 - - [25/May/2015 00:28:51] "GET /services HTTP/1.1" 200 -
127.0.0.1 - - [25/May/2015 00:28:51] "GET /services/test HTTP/1.1" 200 -
127.0.0.1 - - [25/May/2015 00:28:51] "GET /services/test/127.0.0.1:345 HTTP/1.1"
    404 -
```

Листинг 5: Сообщения системы мониторинга в ходе юнит-теста

3.5. Фронтэнд пользователей

Реализация

Тестирование

3.6. Фронтэнд вычислительных узлов

Реализация

Тестирование

3.7. Система управления сессией

Реализация

Тестирование

3.8. Система управления

Реализация

Тестирование

3.9. Система хранения данных

Реализация

Тестирование

3.10. Система балансировки нагрузки

Реализация

Тестирование

3.11. Система вычисления

Реализация

Тестирование

3.12. Вывод

4. Заключение

5. Список литературы