

Выпускная квалификационная работа

РАЗРАБОТКА ТРЕБОВАНИЙ К ИНТЕГРАЦИИ МЕЖДУ СИСТЕМОЙ SERVICE DESK И СИСТЕМОЙ УЧЕТА ЗАДАЧ JIRA

Котельникова Елена Леонидовна

Руководитель: Степчева З.В.

Системный анализ



Обзор предметной области

На данный момент:

- В компании осуществляется техническая поддержка пользователей по работе в системе в полуавтоматическом режиме посредством разных программ: часть пользователей работает в Service Desk, задачи разработчикам в Jira заносятся вручную специалистом, ответ по заявке пользователя осуществляется также вручную специалистом по почте и/или телефону. Программное обеспечение не связано между собой, вследствие чего происходит несинхронность работы служб.

Проблема:

- В связи с ростом заявок обработка в полуавтоматическом режиме становится затруднительной.

Следствие из проблемы:

- Несогласованность работы различных служб между собой;
- Сложно контролировать загрузку подразделения разработчиков в целом и каждого сотрудника в отдельности;
- Снижение лояльности пользователей и оборотов продаж программного обеспечения

Решение:

- Интеграция системы Service Desk со сторонними сервисами.

Цель

Целью данной работы является разработка требований к интеграции системы Service Desk и системы учета задач Jira.

Задачи:

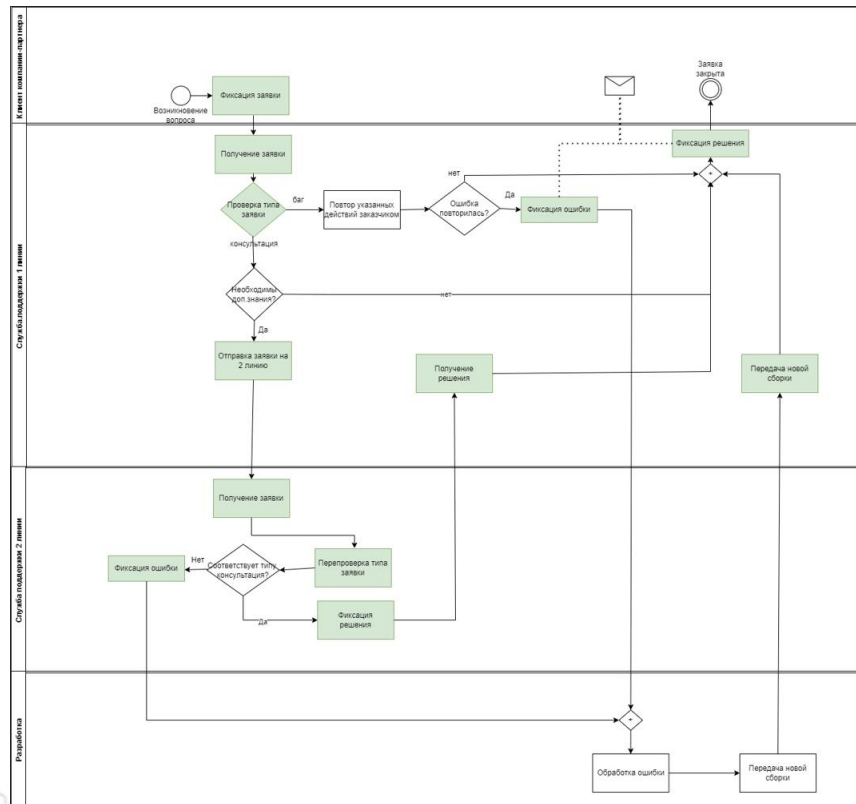
1. Выявить требования к проекту;
2. Разработать модель бизнес-процесса (AS IS и TO BE);
3. Разработать диаграмму потоков данных;
4. Разработать сценарии интеграции;
5. Описать применяемые технологии передачи данных;
6. Разработать диаграмму последовательности;
7. Разработать диаграмму состояний;
8. Разработать описание технологии передачи данных;
9. Описать REST API на примере Регистрация заявки в SD;
10. Анализ результатов и подведение итогов;

Пользовательские требования

Код требования	Требование
Заявка.SD	SD осуществляет регистрацию заявки и передает данные в Jira
Статус.SD	SD осуществляет смену статуса
Приоритет.SD	SD осуществляет смену приоритета
Статус.SD.J	SD забирает статус задачи из Jira
Приоритет.SD.J	SD забирает приоритет задачи из Jira
Данные.SD.USend	SD отправляют данные в Unisender
Заявка.J	Jira получает заявку в качестве задачи
Статус.J	Jira меняет статусы задач
Приоритет.J	Jira меняет приоритеты задач
Данные.USend	Unisender отправляет данные

Схемы бизнес-процессов

AS IS (без применения интеграции)



Схемы бизнес-процессов

ТО VE (с применением интеграции)

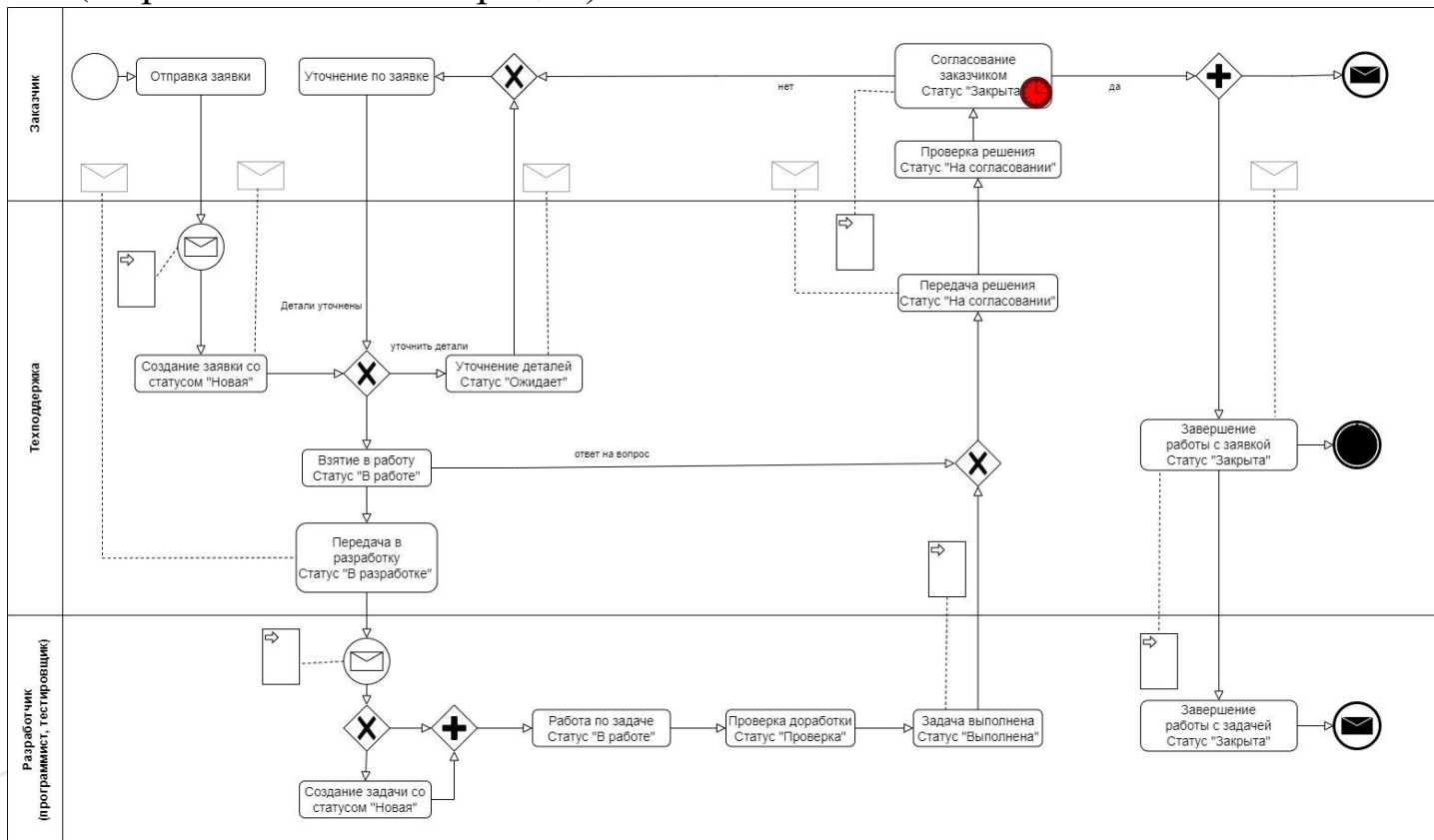
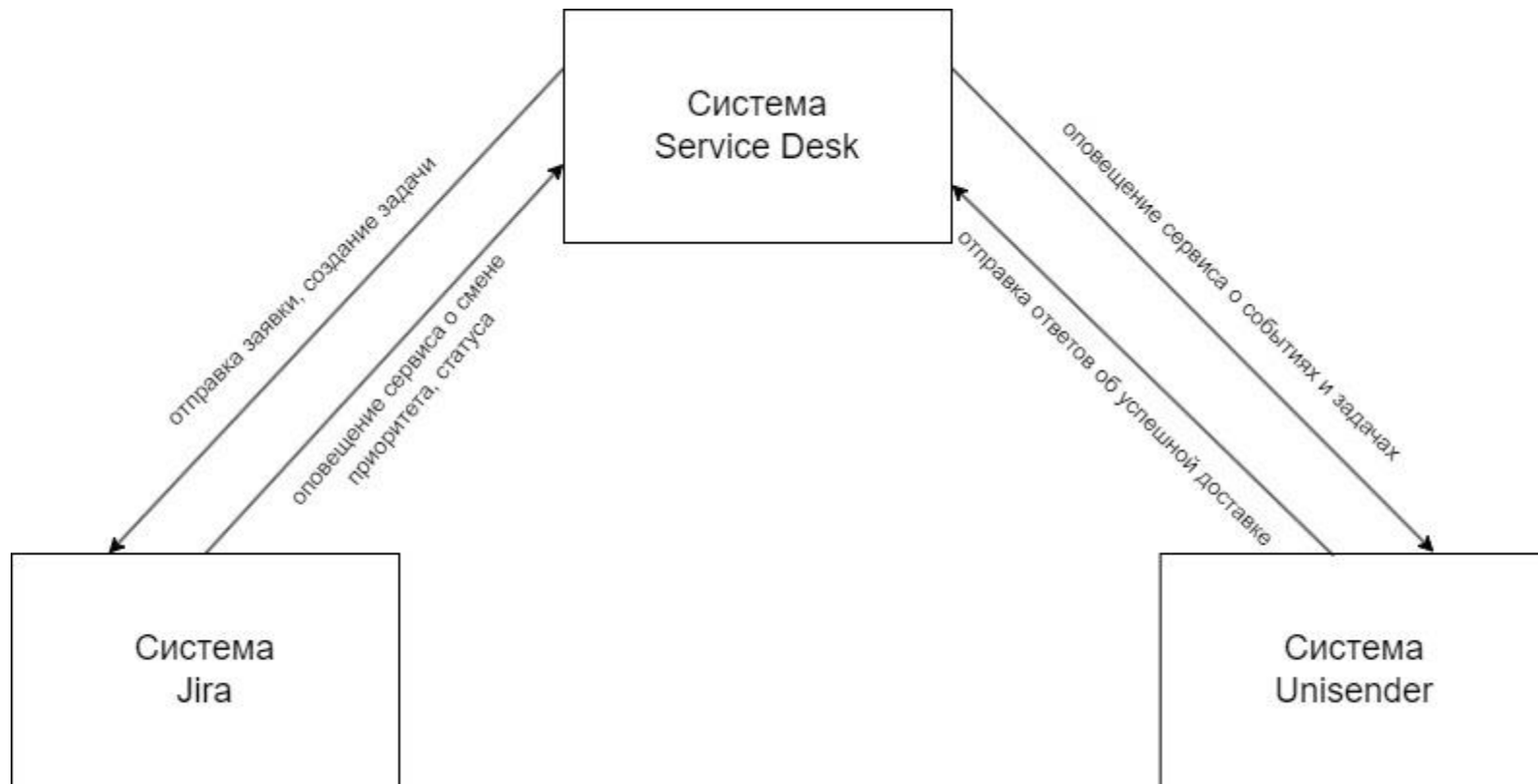


Диаграмма потоков данных между системами



Сценарий интеграции «Передача данных для создания новой заявки в SD»

ID: UC-1.1
Название: Передача данных для создания новой заявки в SD
Предусловия: Пользователь (заказчик) имеет аккаунт, авторизован в системе, заполнил данные заявки.
Триггер: Пользователь нажал кнопку "Отправить заявку".
Постусловия: 1. В систему Usend отправлен номер созданной заявки 2. В системе заявок SD опубликована заявка на обслуживание
Основной поток: 1. SD получает данные пользователя и заявки. 2. SD создает заявку и сохраняет ее. 3. SD <u>журналирует</u> созданную заявку на обслуживание 4. SD передает в систему Usend информацию о полученной заявке 5. Usend направляет уведомление пользователю о созданной заявке

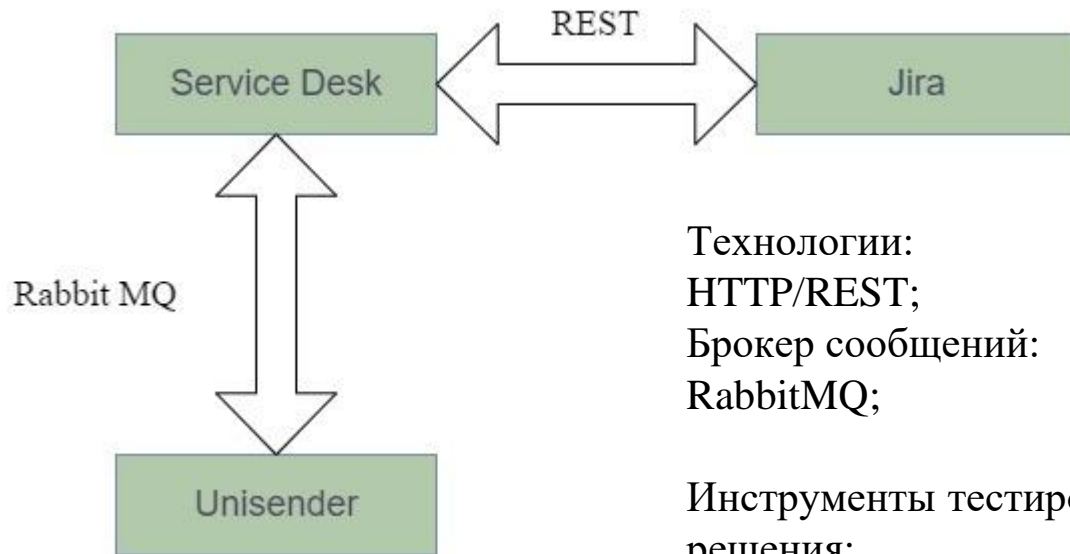
Расширение:

2. SD отвечает пользователю ошибкой валидации введенных данных по заявке.

2a1. В журнал публикуется событие об ошибке валидации данных заявки

Пользователь получает сообщение о невозможности отправить заявку.

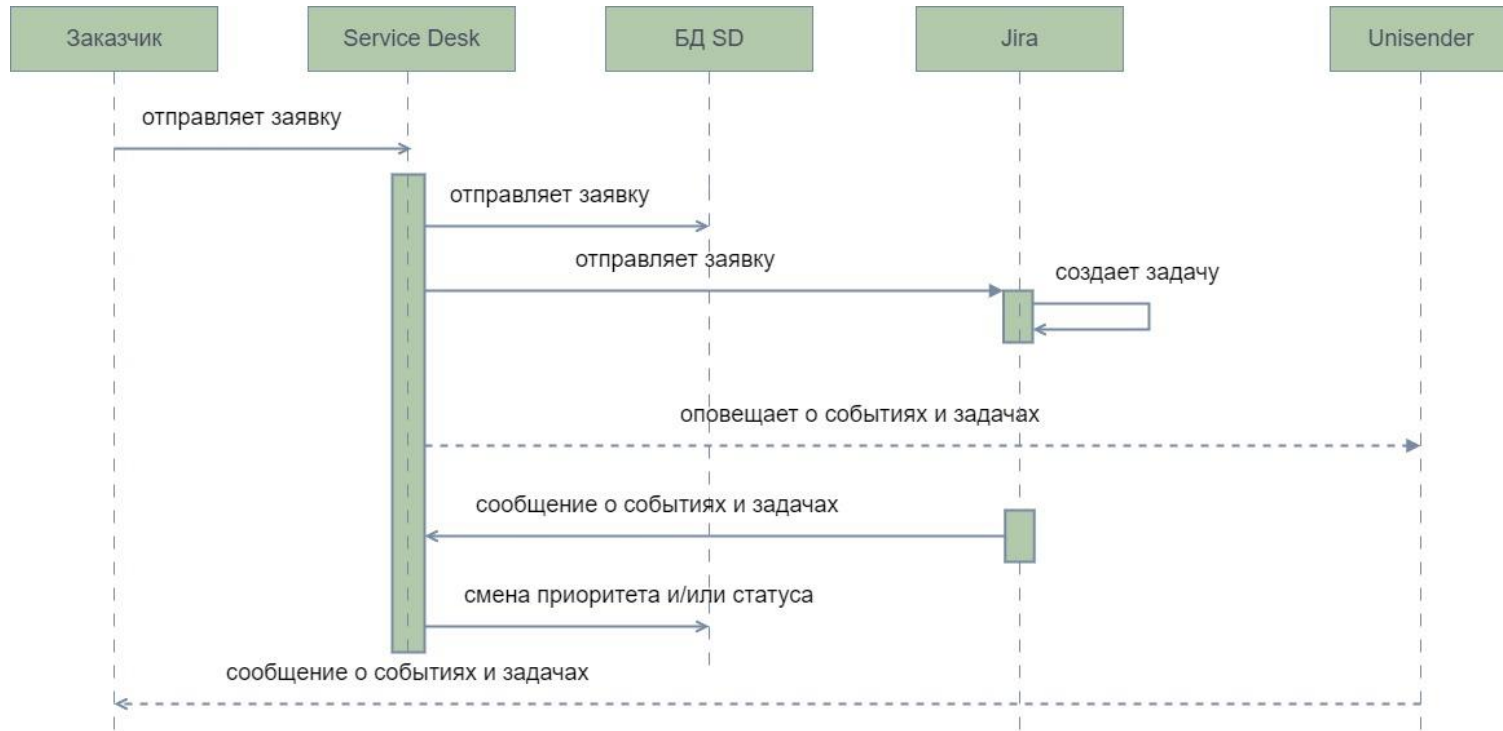
Применяемые к системам технологии передачи данных



Технологии:
HTTP/REST;
Брокер сообщений:
RabbitMQ;

Инструменты тестирования и документирования
решения:
PostMan
Swagger
Конвертеры данных: JSON Editor, JSON Scheme

Диаграмма последовательности



Описание технологии передачи данных

Краткое описание метода: REST API

Сервис предназначен для создания новой заявки в системе SD, по данным, отправленным от пользователя (заказчика).

Метод запроса: POST

Формат ответа: [https://\[host\]:\[post\]/1/Claims](https://[host]:[post]/1/Claims)

Входные параметры:

Параметр	Описание	Тип данных	Обязательность
Title	Тема заявки	Строка	+
Type	Тип заявки	Целое	+
Priority	Приоритет заявки	Список значений	+
Description	Содержание заявки	Строка	+
Files	Файлы	Массив файлов	-

Описание технологии передачи данных

Краткое описание метода: REST API

Сервис предназначен для создания новой заявки в системе SD, по данным, отправленным от пользователя (заказчика).

Метод запроса: POST

Формат ответа: [https://\[host\]:\[port\]/1/Claims](https://[host]:[port]/1/Claims)

Входные параметры:

Параметр	Описание	Тип данных	Обязательность
Title	Тема заявки	Строка	+
Type	Тип заявки	Целое	+
Priority	Приоритет заявки	Список значений	+
Description	Содержание заявки	Строка	+
Files	Файлы	Массив файлов	-

Описание технологии передачи данных

Выходные параметры:

При успешном выполнении система вернет код
ответа: 200 OK

Параметр	Описание	Тип данных	Обязательность
Title	Тема заявки	Строка	+
Type	Тип заявки	Целое	+
Priority	Приоритет заявки	Список значений	+
Description	Содержание заявки	Строка	+
Files	Файлы	Массив файлов	-

Описание технологии передачи данных

Выходные параметры (продолжение таблицы):

SecondName	Автор заявки	Ссылка на справочник Users	+
Date	Дата заявки	Дата	+
ClaimID	ID заявки	Целое	+
Status	Статус заявки	Список значений	+
DateChange	Дата изменения	Дата	-
SecondNameSD	Исполнитель (кто решил)	Ссылка на справочник ExecutorsSD	-

Описание технологии передачи данных

Пример формата JSON:

```
{  
  "Claims":  
  {  
    "Title": "Ошибка работы команды",  
    "Type": "Ошибка",  
    "Priority": "Высокий",  
    "Description": "Ошибка по работе команды Вставки",  
    "Files": [1],  
    "SecondName": "Иванов",  
    "Date": "11/1/2023",  
    "ClaimID": 1,  
    "Status": "Новая",  
    "DateChange": "",  
    "SecondNameSD": "Петров"  
  }  
}
```



Описание технологии передачи данных

Схема ответа получения данных при создании новой заявки в системе SD:

```
1 {
2   "definitions": {},
3   "$schema": "http://json-schema.org/draft-07/schema#",
4   "$id": "https://example.com/object1680091405.json",
5   "title": "Root",
6   "type": "object",
7   "required": [
8     "Claims"
9  ],
10  "properties": {
11    "Claims": {
12      "$id": "#root/Claims",
13      "title": "Claims",
14      "type": "object",
15      "required": [
16        "Title",
17        "Priority",
18        "Description",
19        "Files",
20        "SecondName",
21        "Date",
22        "ClaimID",
23        "Status",
24        "DateChange",
25        "SecondNameSD"
26      ],
27    },
28    "properties": {
29      "Title": {
30        "$id": "#root/Claims/Title",
31        "title": "Title",
32        "type": "string",
33        "default": "",
34        "examples": [
35          "Ошибка работы команды"
36        ],
37        "pattern": "^.*$"
38      },
39    },
40    "Type": {
41      "$id": "#root/Claims/Type",
42      "title": "Type",
43      "type": "string",
44      "default": "",
45      "examples": [
46        "Ошибка"
47      ],
48      "pattern": "^.*$"
49    },
50    "Priority": {
51      "$id": "#root/Claims/Priority",
52      "title": "Priority",
53      "type": "string",
54      "default": "",
55      "examples": [
56        "Высокий"
57      ],
58      "pattern": "^.*$"
59    },
60    "Description": {
61      "$id": "#root/Claims/Description",
62      "title": "Description",
63      "type": "string",
64      "default": "",
65      "examples": [
66        "Высокий"
67      ],
68      "pattern": "^.*$"
69    }
70  }
71 }
```

Описание технологии передачи данных

Схема ответа получения данных при создании новой заявки в системе SD (продолжение):

```
65         "Ошибка по работе команды Вставки"
66     ],
67     "pattern": "^.*$"
68 },
69 "Files": {
70     "$id": "#root/Claims/Files",
71     "title": "Files",
72     "type": "array",
73     "default": [],
74     "items":{
75         "$id": "#root/Claims/Files/items",
76         "title": "Items",
77         "type": "integer",
78         "examples": [
79             1
80     ],
81         "default": 0
82     }
83 },
84 "SecondName": {
85     "$id": "#root/Claims/SecondName",
86     "title": "Secondname",
87     "type": "string",
88     "default": "",
89     "examples": [
90         "Иванов"
91     ],
92     "pattern": "^.*$"
93 },
94 "Date": {
95     "$id": "#root/Claims/Date",
96     "title": "Date",
```

Описание технологии передачи данных

Схема ответа получения данных при создании новой заявки в системе SD (продолжение):

```
97         "type": "string",           112     },
98         "default": "",             113     "Status": {
99         "examples": [              114         "$id": "#root/Claims/Status",
100         "11/1/2023"                115         "title": "Status",
101     ],                               116         "type": "string",
102     "pattern": "^.*$"              117         "default": "",
103 },                                   118         "examples": [
104 "ClaimID": {                       119         "Новая"
105     "$id": "#root/Claims/ClaimID"  120     ],
106     "title": "Claimid",            121     "pattern": "^.*$"
107     "type": "integer",              122 },
108     "examples": [                  123 "DateChange": {
109         1                            124     "$id": "#root/Claims/DateChange",
110     ],                               125     "title": "Datechange",
111     "default": 0                     126     "type": "string",
                                       127     "default": "",
```

Описание технологии передачи данных

Схема ответа получения данных при создании новой заявки в системе SD (продолжение):

```
128     "examples": [  
129         ""  
130     ],  
131     "pattern": "^.*$"  
132 },  
133 "SecondNameSD": {  
134     "$id": "#root/Claims/SecondNameSD",  
135     "title": "Secondnamesd",  
136     "type": "string",  
137     "default": "",  
138     "examples": [  
139         "Петров"  
140     ],  
141     "pattern": "^.*$"  
142 }  
143 }  
144 }  
145 }  
146 }  
147 }  
148 }
```

Тестирование и документирование запросов

На рисунке представлен GET запрос поиска заявки по ее ID, выполненный в Swagger. Swagger позволяет исключить ошибки в запросе и подготовить документацию для отдела разработки:

Claim

GET /claim Метод получения заявки по ID

Parameters Try it out

Name	Description
ClaimId * required integer(\$int64) (path)	Вернуть заявку по ее ID

ClaimId

Responses

Тестирование и документирование запросов

GET запрос поиска заявки по ее ID в Swagger (продолжение):

Code	Description	Links
200	OK	No links
<p>Media type</p> <p><input type="text" value="application/json"/> ▾</p> <p>Controls Accept header.</p> <p>Example Value Schema</p> <pre>{ "Title": "Ошибка работы команды", "Description": "Необходимо доработать панель ручного ввода данных", "Type": "Ошибка", "Priority": "Низкий", "SecondNameU": "Иванов", "SecondNameSd": "Петров", "Date": "2023-03-01 15:00:00", "DateChange": "2023-05-01 15:00:00", "ClaimId": 1589, "StatusClaim": "Новая", "Files": "name1, name2" }</pre>		
400	Bad Request	No links
404	Not Found	No links

Тестирование и документирование запросов

GET запрос поиска заявки по ее ID в Swagger (продолжение):

Schemas

```
Claim {
  Title* string
  example: Ошибка работы команды
  Тема заявки (кратко описать суть)

  Description* string
  example: Необходимо доработать панель ручного ввода данных
  Описание тела заявки

  Type* string
  example: Ошибка
  Тип заявки

  Priority* string
  example: Низкий
  Приоритет заявки. Значение по умолчанию - "Низкий"

  SecondNameU* string
  example: Иванов
  Фамилия автора заявки (заказчика)

  SecondNameSd* string
  example: Петров
  Фамилия исполнителя заявки в SD

  Date* string
  Format: 2023-03-01T15:00:00
  example: 2023-03-01 15:00:00
  Дата создания заявки

  DateChange* string
  Format: 2023-05-01T15:00:00
  example: 2023-05-01 15:00:00
  Дата изменения заявки. В случае отсутствия, будет возвращена дата создания.

  ClaimId* {
    description: Идентификатор заявки
  }
  example: 1589

  StatusClaim* string
  example: Новая
  Статус заявки. Для вновь созданной - "Новая"

  Enum:
  [ new, in_operation, in_coding, waiting, on_approval, closed ]

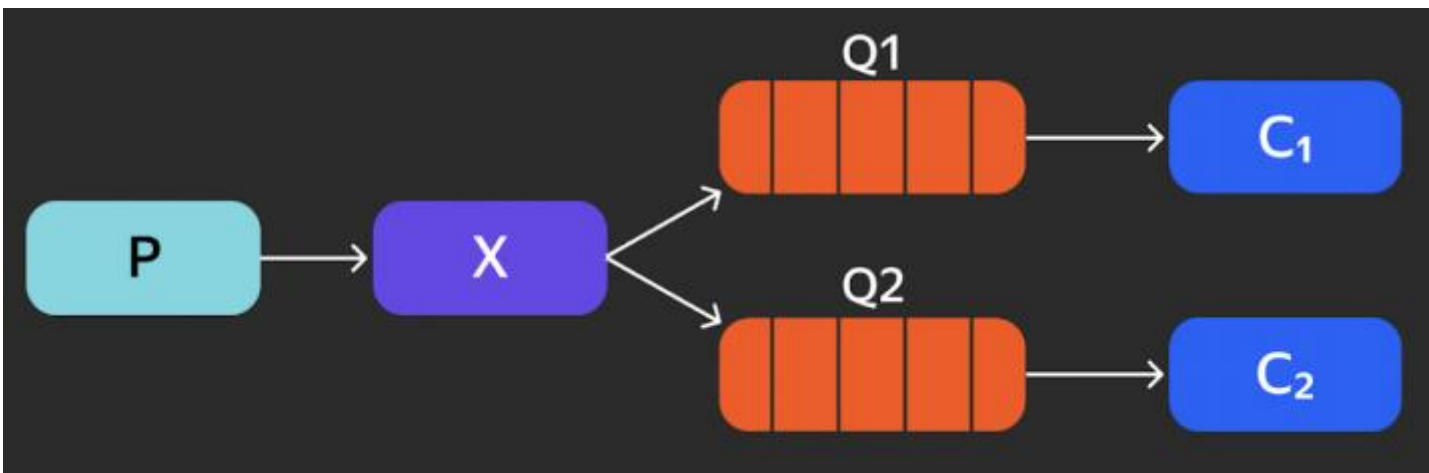
  Files* string
  example: name1, name2
  Передаваемые файлы
}
```

Описание технологии передачи данных

Для передачи данных между Service Desk и Unisender используется RabbitMQ.

Работа Exchange в RabbitMQ: Fanout Exchange (отправка сообщений во все очереди).

Обеспечит оповещение Unisender о событиях и задачах в Service Desk.



Описание технологии передачи данных Rabbit MQ

Метод создания Exchange:

Метод создания обменника:

<pre>//... channel.ExchangeDeclare(exchange: "message", type: "fanout", durable: "true", autoDelete: "false", arguments: null); //...</pre>	<p>Название обменника (exchange) - message Тип обменника (type) - fanout Параметры хранения exchange – true (будет храниться после перезагрузки брокера или сервера) Автоматическое удаление exchange (autoDelete) - будет удален при удалении всех связанных очередей. Аргументы (arguments) — значение null, так как не нужно задавать альтернативный маршрут по другому пути.</p>
---	--

Queues

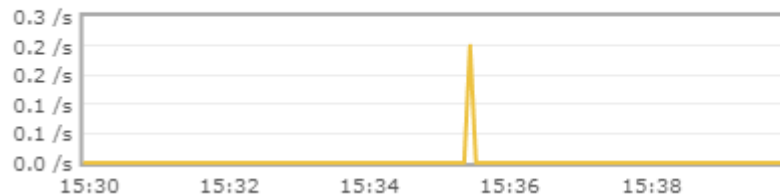
▼ All queues (2)

Pagination

Page 1 of 1 - Filter: Regex ?

Overview				Messages			Message rates			+/-
Name	Type	Features	State	Ready	Unacked	Total	incoming	deliver / get	ack	
email queue	classic	D Lim Args HA <code>уходpnja-max-length</code>	idle	0	0	0				

Message rates last ten minutes ?



Publish 0.00/s

Описание технологии передачи данных Rabbit MQ

Параметры конфигурации очереди:

Устойчивость (долговременность)	durability	да (durable)
Автоматическое удаление	auto delete	нет <input type="checkbox"/>
Повторная публикация сообщений с истекшим TTL и сообщений, отклоненных потребителями	да	
Точка обмена	x-dead-letter-exchange	messenger
Ключ маршрутизации	x-dead-letter-routing-key	-
Ограничение TTL на очередь	x-expires	нет
Ограничение TTL сообщений на каждую очередь, мс	x-message-ttl	60000

Описание технологии передачи данных Rabbit MQ

Параметры конфигурации очереди (продолжение таблицы):

Ограничение при переполнении очереди	x-overflow	reject-publish
Максимальное количество сообщений в очереди	x-max-length	500
ASK	да	

Описание технологии передачи данных Rabbit MQ

Публикация сообщения в Rabbit MQ для отправки уведомления пользователю о создании заявки:

```
{  
  "Service Desk": "Claims"  
  {  
    "UserID": "25894358",  
    "SecondName": "Иванов",  
    "Date": "2023-03-15",  
    "ClaimID": 1,  
    "Status": "Новая",  
    "Message": "Ваша заявка с номером {ClaimID} отправлена",  
    "Mail": "Ivanov@yandex.ru"  
  }  
}
```

Анализ полученных результатов

В ходе работы над проектом получены следующие результаты:

1. Определены и описаны цели и задачи проекта.
2. Разработаны модели бизнес-процессов (AS IS и TO BE).
4. Разработана модель данных.
5. Разработана диаграмма последовательности.
6. Разработаны и представлены сценарии использования (use case).
7. Разработан технический проект обмена данными между Service Desk и Jira
8. Изучены и использованы следующие инструменты:

Postman <https://web.postman.co/>

Swagger <https://editor.swagger.io/>

Drawio <https://app.diagrams.net/>

PlantText <https://www.planttext.com/>

Json Editor Online <https://jsoneditoronline.org>

Rabbit MQ <https://porpoise.rmq.cloudamqp.com/#/channels>

Extends Class <https://extendsclass.com/json-schema-validator.html>

API JIRA <https://docs.atlassian.com/software/jira/docs/api/REST/7.6.1>

Построение диаграмм последовательности <https://www.websequencediagrams.com/>

