ВШЭ и НСПК

Mir Geo API

Москва, 2020

Оглавление

1. История изменений

№ версии	Дата обновления	Краткое изменение описания	Обновлено
0.0.1	01.03.2021	UML use cases	Мурашкин М.
0.0.2	15.03.2021	Examples of requests and responses UML use cases update Data Models	Мурашкин М.

2. Mir Geo. Введение

2.1. Назначение и содержание

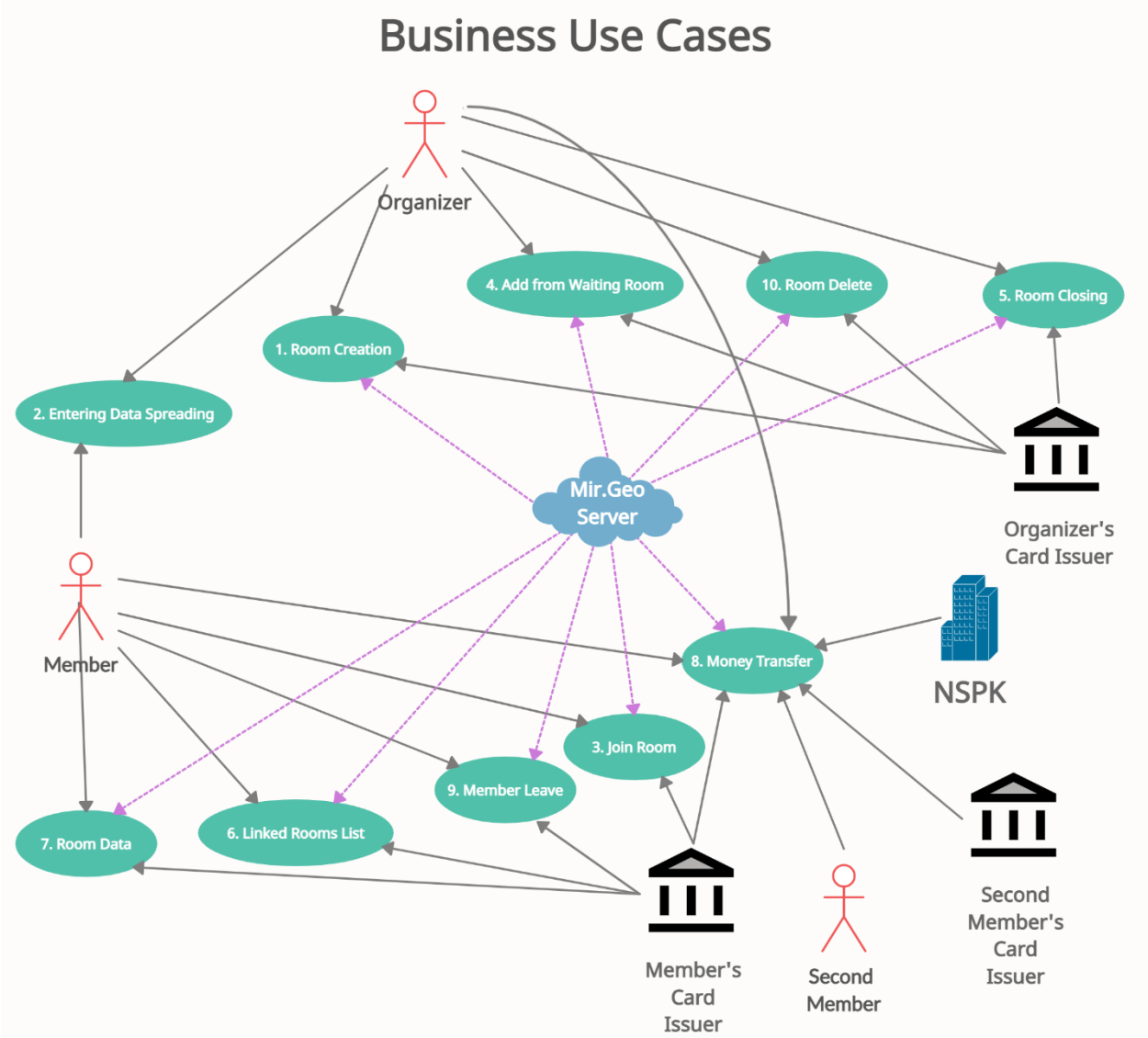
Представленный документ исчерпывающе описывает “общение” с сервисом, интерфейс, внешнее представление, какие элементы бизнес данных задействованы в информационных потоках, а также логику внутреннего устройства и правила взаимодействия компонентов.

2.2. Описание идеи

Идея сервиса **—** реализовать возможность переводить или запрашивать деньги внутри группы людей, территориально находящейся в одном месте, новым удобным способом. Идея основана на механизме создания комнат, привязанных к определенным местоположениям.

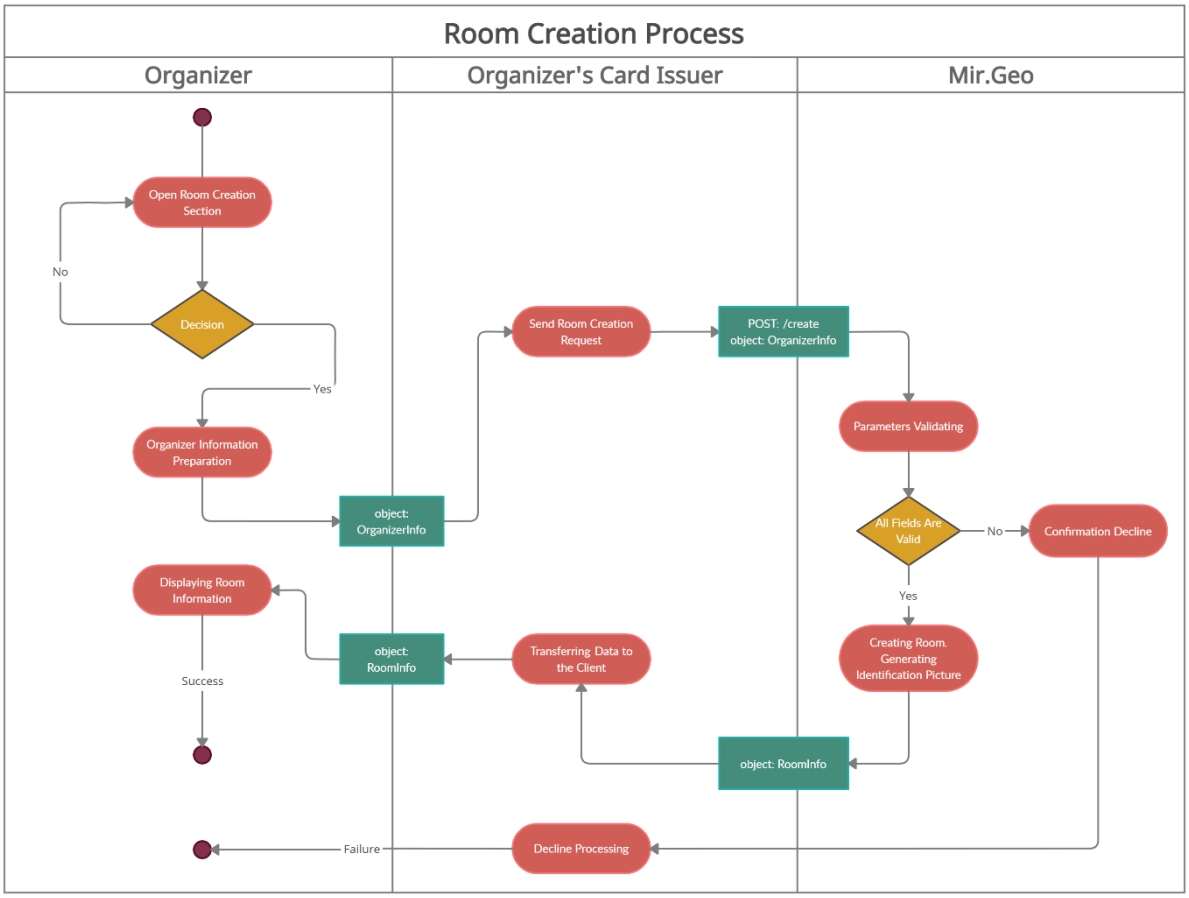
3. Концептуальная архитектура сервиса

3.1 Бизнес-архитектура сервиса



3.2. Компоненты: функции и участие в информационных потоках

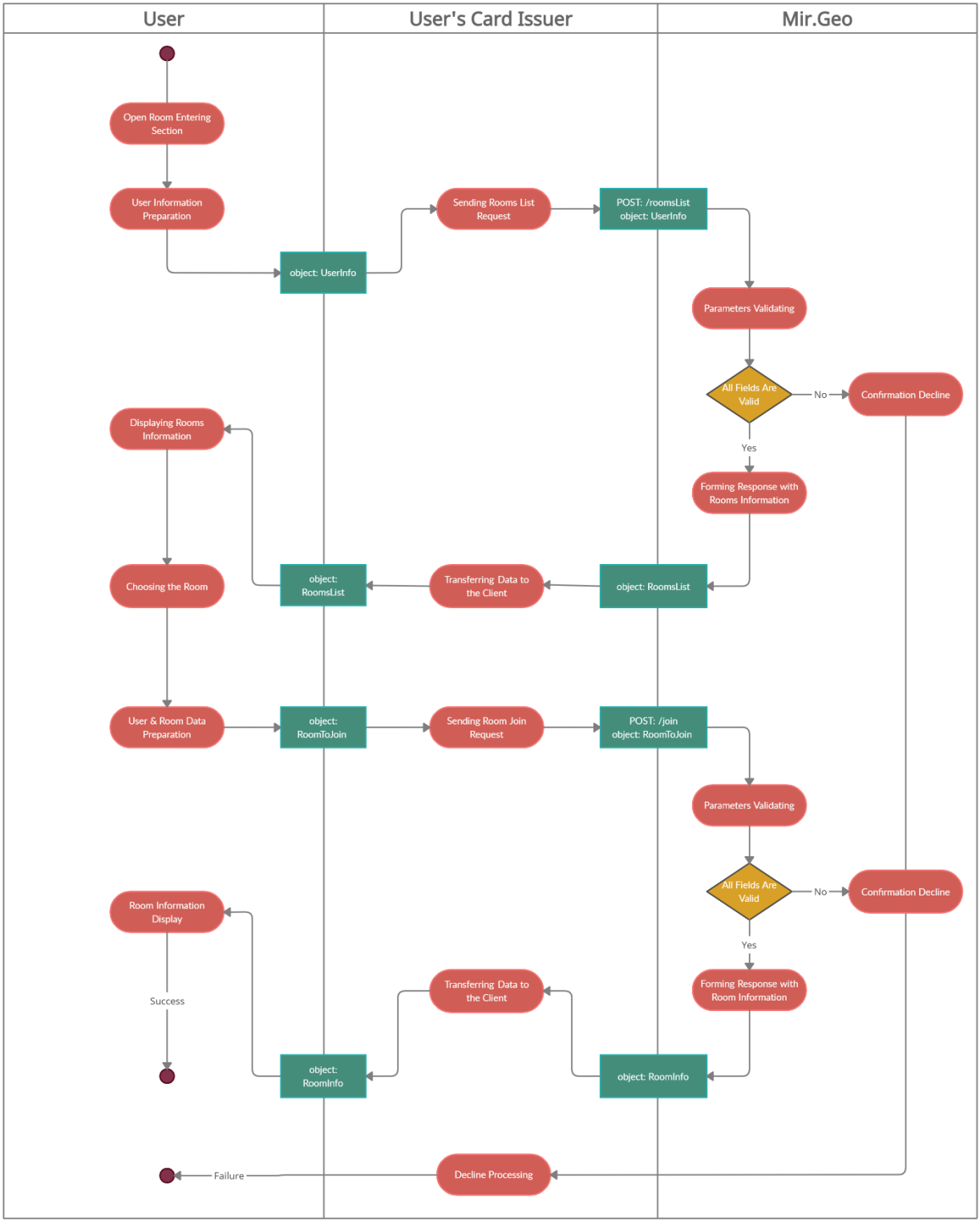
3.2.1 Взаимодействие компонентов при создании комнаты



Процесс создания комнаты

Вся информация, полученная от клиента, валидируется на сервере Mir Geo (кроме данных по типу номера счета и идентификатора банка).

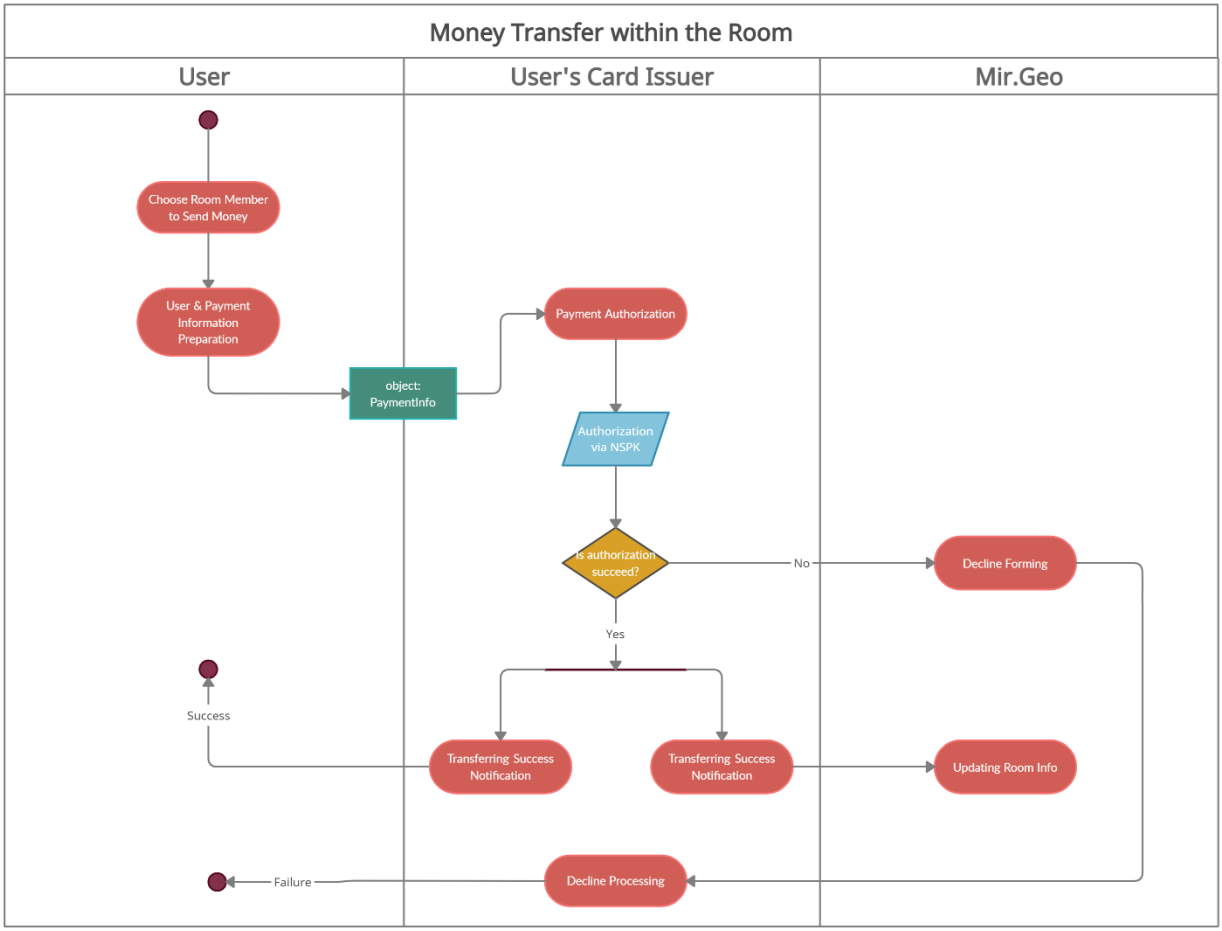
3.2.2 Взаимодействие компонентов при подключении пользователя к созданной комнате



Процесс подключения пользователя к комнате

Вся информация, полученная от клиента, валидируется на сервере Mir Geo (кроме данных по типу номера счета и идентификатора банка).

3.2.3 Взаимодействие компонентов при подключении пользователя к созданной комнате



Процесс перевода средств внутри комнаты

3.2.4. Мобильное приложение банка

Данный компонент реализует интерфейс процесса создания комнаты, отображения ее содержимого и перевода денег между ее участниками. Банковское приложение может “общаться” только с компонентом Банк, который в свою очередь отправляет запросы на сервер Mir Geo.

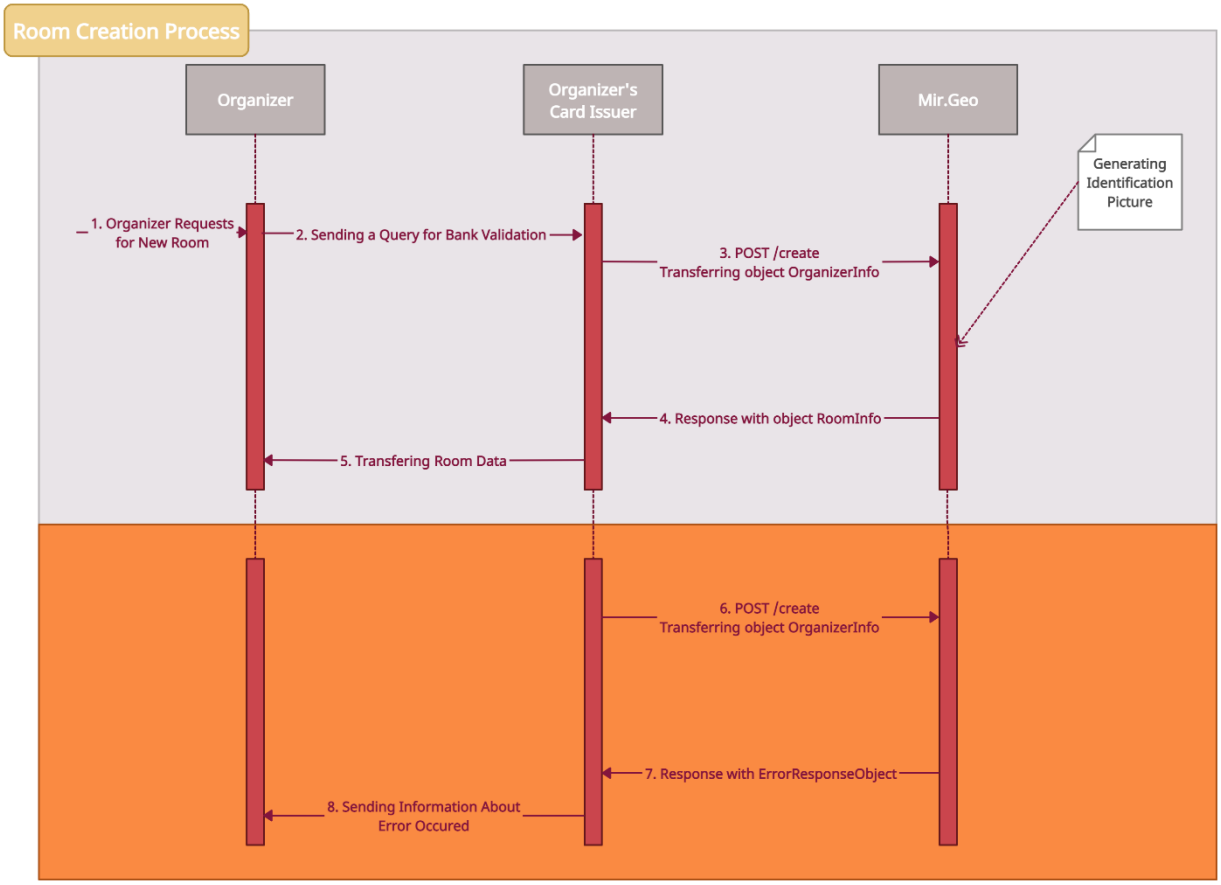
3.2.5. Банк

Является связующим компонентом между мобильным приложением и сервером Mir Geo. Осуществляет передачу и получение объектов UserInfo, OrganizerInfo, RoomInfo, RoomsList и RoomToJoin, производит валидацию полученной от приложения информации, отправляет запросы на авторизацию банковского перевода. Заметим, что все запросы от Мобильного приложения не могут обойти компонент Банк и напрямую обратиться к серверу Mir Geo.

3.2.6. Mir Geo

Этот компонент получает запросы только от Банка, реализует REST API для создания, редактирования и получения информации о комнатах, а также взаимодействия с ними.

3.2.7. Информационные потоки: назначение, состав и участвующие компоненты



Процесс создания комнаты

Шаг	Описание сообщения
Поток 1: Room Creation Process	
1	Клиент делает запрос на создание новой комнаты
2	Запрос в банк на валидацию данных клиента
3	Запрос в сервис на создание новой комнаты, по заданным параметрам
4	Ответ от сервиса, содержащий результат запроса 3 и информацию о комнате
5	Передача результата пользователю
6	Запрос в сервис на создание новой комнаты, по заданным параметрам
7	Ответ от сервиса, описывающий ошибку
8	Передача результата пользователю

3.3. Описание элементов бизнес-данных

Элемент бизнес-данных	Название в API	Описание
PAM	PAM	Совокупность данных, идентифицирующих пользователя
Имя получателя	PAM.name	Фамилия, имя и отчество получателя
Счет получателя	PAM.accountNumber	Номер счета получателя
Номер банка	PAM.bankIdentificationNumber	Уникальный номер банка получателя
Список участников	memberList	Список участников комнаты и информация о них
Организатор	organizer	Организатор комнаты
Изображение	uniquePicture	Картинка, идентифицирующая комнату
Переводы	transfersData	Информация о переводах, совершенных внутри комнаты
Состояние	status	Одно из двух состояний комнаты – открытая или закрытая

4. Требования к пользовательскому интерфейсу

4.1. Пользовательские сценарии

4.1.1. User script: Создание комнаты

Базовый сценарий:

- Пользователь переходит в приложение своего банка, открывает раздел для создания комнат.
- Выбирает необходимые параметры (наличие зала ожидания, местоположение комнаты).
- Нажимает кнопку создать комнату и в ответ получает уникальную картинку, которую необходимо показать будущим участникам комнаты.

4.1.2. User script: Подключение к комнате

Предусловие: наличие картинки, идентифицирующей комнату
Базовый сценарий:

- Пользователь переходит в приложение своего банка, открывает раздел для подключения к созданной комнате.
- Получает список комнат, созданных поблизости него, и уникальные картинки каждой из них.
- Выбирает нужную ему и подключается. Попадает в комнату и получает подробную информацию о ней (список участников и ранее совершенные переводы внутри нее)

4.1.3. User script: Перевод средств

Предусловие: пользователь подключен к комнате
Базовый сценарий:

- Пользователь открывает список участников комнаты.
- Выбирает участника комнаты, которому хочет перевести деньги.
- Указывает сумму для перевода и нажимает отправить.

4.2. Требования к интерфейсу

При внедрении сервиса в банковское приложения, нужно следовать следующим рекомендациям:

1) На экране со списком способов перевода должен быть добавлен метод перевода “подключение к комнате”
2) На экране создания комнате должна присутствовать возможность выбрать необходимые параметры создания (наличие зала ожидания, местоположение комнаты, смена сгенерированной картинки на новую)
3) После успешного создания комнаты приложение должно отобразить только что созданную комнату и сгенерированную картинку
4) Предоставить пользователю возможность просматривать свои комнаты, к которым он привязан, а также возможность выходить из них и редактировать, если он является организатором

5. Требования к безопасности

5.1. Безопасность мобильного клиента банка

Сервис будет встраиваться в банковское приложение, поэтому необходимо обеспечить защищенное соединение с сервером банковского приложения. Также приложение банка должно соблюдать все правила безопасности во время разработки для защиты от взлома “изнутри” (обеспечение защиты от reverse engineering, от встраивания вирусов).

5.2. Безопасность доступа к методам API

Необходимо организовать защищенное зашифрованное соединение между банком и сервисом (пример - TLS 1.2).

6. Требования к прикладному транспорту сообщений

HTTP Operation	Endpoint	Scope	Message Signing	Grant Type	Idem. Key	Request Object	Response Object
POST	POST /create	Нет	Нет	Authorization Token	Да	OrganizerInfo	RoomInfo
POST	POST /roomList	Нет	Нет	Authorization Token	Нет	UserInfo	RoomList
POST	POST /join	Нет	Нет	Authorization Token	Нет	RoomToJoin	RoomInfo
POST	POST /addUser	Нет	Нет	Authorization Token	Нет	AddUser	N/A
POST	POST /myRooms	Нет	Нет	Authorization Token	Нет	UserInfo	RoomList
DELETE	DELETE /deleteRoom	Нет	Нет	Authorization Token	Нет	RoomInfo	N/A
DELETE	DELETE /deleteUser	Нет	Нет	Authorization Token	Нет	UserInfo	N/A

6.1.1. POST /create

Метод POST, применяемый к эндпоинту /create, позволяет Банку запрашивать у сервиса Mir Geo создание новой комнаты. В запросе Банк должен передать информацию OrganizerInfo.

6.1.1.1. Требования к заголовкам запроса

Header	Обязательный	Описание
AuthToken	Да	Токен авторизации Банка
x-Idempotency-Key	Да	Ключ идемпотентности

- Действие POST указывает сервису Mir Geo, что комната должна быть создана.
- В результате действия POST компонент Mir Geo создает комнату и отвечает RoomInfo, содержащим идентифицирующую картинку, необходимую будущим участникам для определения нужной им команты.

6.1.1.2. Возможные ответы (Responses)

При успешном создании комнаты:

- Response (HTTP Code: 200 OK, Body: RoomInfo)

Возможные ошибки во время запроса:

- Response (HTTP Code: 400 Bad Request, Body: ErrorResponseObject) - при неверных переданных параметрах.
- Response (HTTP Code: 401 Unauthorized, Body: ErrorResponseObject) - при неверном переданном токене авторизации.

6.1.2. POST /roomList

Метод POST, применяемый к эндпоинту /roomList, позволяет Банку запрашивать у сервиса Mir Geo список комнат, соответствующих переданной пользователем геолокации.

6.1.2.1. Требования к заголовкам запроса

Header	Обязательный	Описание
AuthToken	Да	Токен авторизации Банка
x-Idempotency-Key	Нет	Ключ идемпотентности

- Действие POST указывает сервису Mir Geo, что нужно найти в базе данных на сервере список подходящих комнат и отправить информацию о них ответом на запрос.
- В результате действия POST компонент Mir Geo передает список всех подходящих комнат с их уникальными изображениями.

6.1.2.2. Возможные ответы (Responses)

При успешном нахождении подходящих комнат:

- Response (HTTP Code: 200 OK, Body: RoomList)

Возможные ошибки во время запроса:

- Response (HTTP Code: 400 Bad Request, Body: ErrorResponseObject) - при неверных переданных параметрах.
- Response (HTTP Code: 401 Unauthorized, Body: ErrorResponseObject) - при неверном переданном токене авторизации.

6.1.3. POST /join

Метод POST, применяемый к эндпоинту /join, позволяет Банку отправлять запрос на сервер Mir Geo на подключение пользователя к комнате.

6.1.3.1. Требования к заголовкам запроса

Header	Обязательный	Описание
AuthToken	Да	Токен авторизации Банка
x-Idempotency-Key	Нет	Ключ идемпотентности

- Действие POST указывает сервису Mir Geo, что нужно присоединить пользователя, отправившего запрос, к комнате.
- В результате действия POST компонент Mir Geo подключает пользователя к комнате и обновляет информацию о ней.

6.1.3.2. Возможные ответы (Responses)

При успешном подключении к комнате:

- Response (HTTP Code: 200 OK, Body: RoomInfo)

Возможные ошибки во время запроса:

- Response (HTTP Code: 400 Bad Request, Body: ErrorResponseObject) - при неверных переданных параметрах.
- Response (HTTP Code: 401 Unauthorized, Body: ErrorResponseObject) - при неверном переданном токене авторизации.

6.1.4. POST /myRooms

Метод POST, применяемый к эндпоинту /myRooms, позволяет Банку отправлять запрос на сервер Mir Geo на получение списка всех комнат, привязанных к пользователю.

6.1.4.1. Требования к заголовкам запроса

Header	Обязательный	Описание
AuthToken	Да	Токен авторизации Банка
x-Idempotency-Key	Нет	Ключ идемпотентности

- Действие POST указывает сервису Mir Geo, что нужно в базе данных найти список комнат, участником или организатором которых является пользователь.
- В результате действия POST компонент Mir Geo находит возвращает список таких комнат.

6.1.4.2. Возможные ответы (Responses)

При успешном нахождении комнат:

- Response (HTTP Code: 200 OK, Body: RoomsList)

Возможные ошибки во время запроса:

- Response (HTTP Code: 400 Bad Request, Body: ErrorResponseObject) - при неверных переданных параметрах.
- Response (HTTP Code: 401 Unauthorized, Body: ErrorResponseObject) - при неверном переданном токене авторизации.

6.1.5. DELETE /deleteRoom

Метод POST, применяемый к эндпоинту /deleteRoom, позволяет Банку удалить определенную комнату.

6.1.5.1. Требования к заголовкам запроса

Header	Обязательный	Описание
AuthToken	Да	Токен авторизации Банка
x-Idempotency-Key	Нет	Ключ идемпотентности

- Действие DELETE указывает сервису Mir Geo, что нужно сделать созданную ранее комнату неактивной.
- В результате действия DELETE указанная комната перестает быть доступной пользователям и отправляется в архив.

6.1.5.2. Возможные ответы (Responses)

При успешном удалении комнаты:

- Response (HTTP Code: 200 OK, Body: RoomsList)

Возможные ошибки во время запроса:

- Response (HTTP Code: 400 Bad Request, Body: ErrorResponseObject) - при неверных переданных параметрах.
- Response (HTTP Code: 401 Unauthorized, Body: ErrorResponseObject) - при неверном переданном токене авторизации.

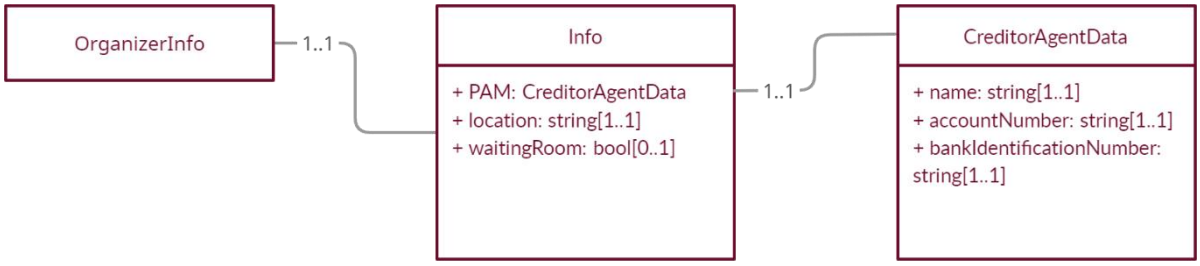
6.2. Модель данных

6.2.1. OrganizerInfo

Объект класса OrganizerInfo, состоит из секции:

- info, который представлен следующими входными элементами:
  - PAM, который представлен объектом CreditorAgentData
  - location
  - waitingRoom

6.2.1.1. UML – диаграмма

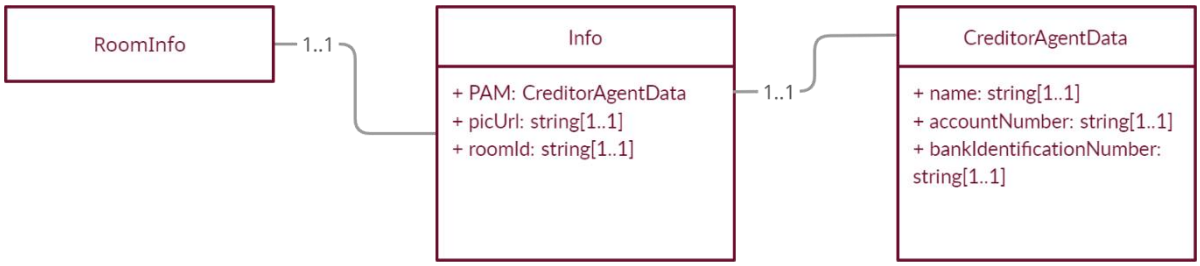


6.2.2. RoomInfo

Объект класса RoomInfo, состоит из секции:

- info, который представлен следующими входными элементами:
  - PAM, который представлен объектом CreditorAgentData
  - picUrl
  - roomId

6.2.2.1. UML – диаграмма

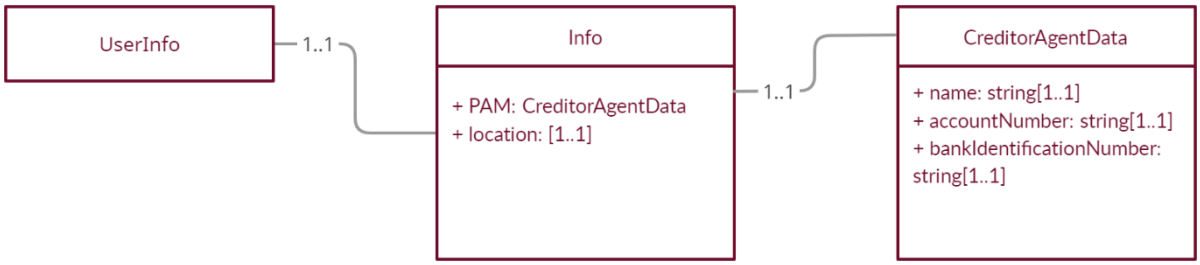


6.2.3. UserInfo

Объект класса UserInfo, состоит из секции:

- info, который представлен следующими входными элементами:
  - PAM, который представлен объектом CreditorAgentData
    location

6.2.3.1. UML – диаграмма

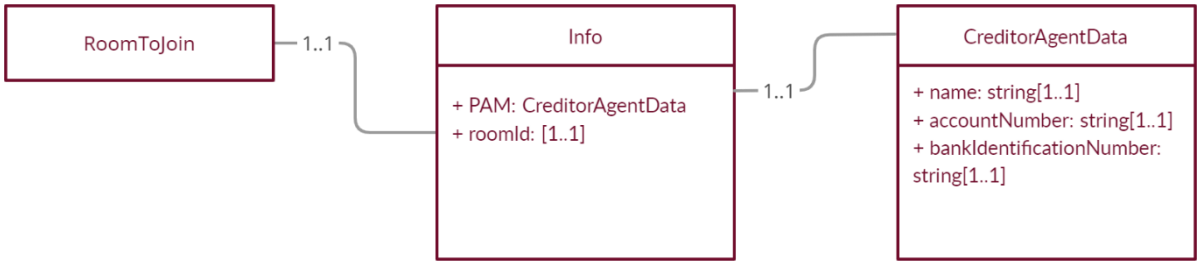


6.2.4. RoomToJoin

Объект класса RoomInfo, состоит из секции:

- info, который представлен следующими входными элементами:
  - PAM, который представлен объектом CreditorAgentData
  - roomId

6.2.4.1. UML – диаграмма

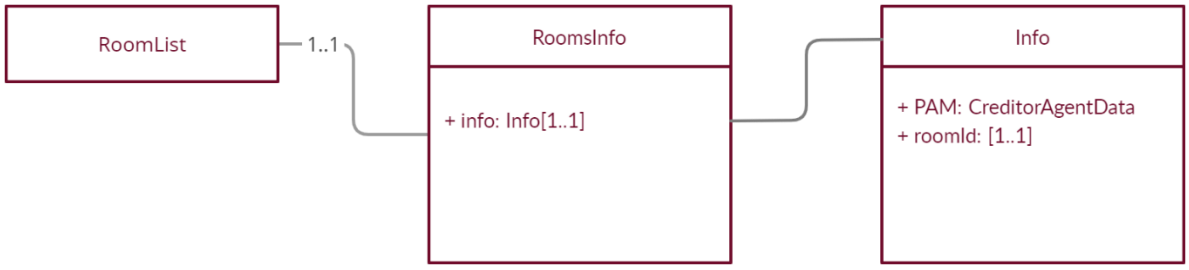


6.2.5. RoomList

Объект класса RoomList, состоит из секции:

- info, состоящего из массива объектов со следующими данными:
  - picUrl
  - roomId

6.2.5.1. UML – диаграмма



6.3. Примеры использования

6.3.1. POST /create

Room Creation Request

```
POST /create HTTP/1.1
AuthToken: npVykVmm5XPPoDgFrVGM
x-Idempotency-Key: nTMVKEGdeIcCMKRItAmumfQcbudpuG
Content-Type: application/json
Content-Length: 255
{
"info" : {
  "PAM" : {
    "name" : "Мурашкин Михаил Павлович",
    "accountNumber" : "41413494875561104839",
    "bankIdentificationNumber" : "5199942"
  },
  "location" : "52.37554296022394, 4.8890930323825375",
  "waitingRoom" : "True"
  }
}
```

Room Creation Response

```
HTTP/1.1 201 Created
Server: nginx/1.10.3
Date: Mon, 15 Mar 2021 15:32:37 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 28
{
 "info": {
    "picUrl" : "mir.geo/id-pictures/ehIgDwegxX4vdhDg",
    "roomId" : "gf2SgxX4vdhDg"
  }
}
```

6.3.2. POST /roomList

Rooms Nearby Request

```
POST /roomList HTTP/1.1
AuthToken: npVykVmm5XPPoDgFrVGM
Content-Type: application/json
Content-Length: 255
{
"info" : {
  "PAM" : {
    "name" : "Мурашкин Михаил Павлович",
    "accountNumber" : "41413494875561104839",
    "bankIdentificationNumber" : "5199942"
  },
  "location" : "52.37554296022394, 4.8890930323825375",
  }
}
```

Rooms Nearby Response

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Mon, 15 Mar 2021 15:32:37 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 100
{
 "info": [
    {
      "picUrl" : "mir.geo/id-pictures/1ehIgDwegxX4vdhDg",
      "roomId" : "1gf2SgxX4vdhDg"
    },
    {
      "picUrl" : "mir.geo/id-pictures/2ehIgDwegxX4vdhDg",
      "roomId" : "2gf2SgxX4vdhDg"
    },
    {
      "picUrl" : "mir.geo/id-pictures/3ehIgDwegxX4vdhDg",
      "roomId" : "3gf2SgxX4vdhDg"
    }
 ]
}
```

6.3.3. POST /join

Room Enter Request

```
POST /join HTTP/1.1
AuthToken: npVykVmm5XPPoDgFrVGM
Content-Type: application/json
Content-Length: 255
{
"info" : {
  "PAM" : {
    "name" : "Мурашкин Михаил Павлович",
    "accountNumber" : "41413494875561104839",
    "bankIdentificationNumber" : "5199942"
  },
  "roomId" : "n7wfykVmasddm5XPPoDgFrVGM",
  }
}
```

Room Enter Response

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Mon, 15 Mar 2021 15:32:37 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 100
{
 "roomId" : "1gf2SgxX4vdhDg"
 "members": [
    {
      "PAM": {
        "name": "Мурашкин Михаил Павлович",
        "accountNumber": "41413494875561104839",
        "bankIdentificationNumber" : "5199942"
      },
      status: "organizer"
    },
    {
      "PAM": {
        "name": "Мурашкин Павел Павлович",
        "accountNumber": "41413494875561104839",
        "bankIdentificationNumber" : "5199942"
      },
      status: "member"
    },
    {
      "PAM": {
        "name": "Мурашкин Михаил Михаилович",
        "accountNumber": "41413494875561104839",
        "bankIdentificationNumber" : "5199942"
      },
      status: "member"
    }
 ]
}
```

6.3.4. POST /myRooms

All Rooms Request

```
POST /myRooms HTTP/1.1
AuthToken: npVykVmm5XPPoDgFrVGM
Content-Type: application/json
Content-Length: 255
{
"info" : {
  "PAM" : {
    "name" : "Мурашкин Михаил Павлович",
    "accountNumber" : "41413494875561104839",
    "bankIdentificationNumber" : "5199942"
  },
  }
}
```

All Rooms Response

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Mon, 15 Mar 2021 15:32:37 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 100
{
 "rooms": [
    {
      "roomId" : "1gf2SgxX4vdhDg"
      "status": "organizer"
    },
    {
      "roomId" : "2gf2SgxX4vdhDg"
      "status": "member"
    },
    {
      "roomId" : "3gf2SgxX4vdhDg"
      "status": "organizer"
    },
 ]
}
```

6.3.5. DELETE /deleteRoom

Delete Room Request

```
DELETE /deleteRoom HTTP/1.1
AuthToken: npVykVmm5XPPoDgFrVGM
Content-Type: application/json
Content-Length: 255
{
"info" : {
  "PAM" : {
    "name" : "Мурашкин Михаил Павлович",
    "accountNumber" : "41413494875561104839",
    "bankIdentificationNumber" : "5199942"
  },
  "roomId" : "n7wfykVmasddm5XPPoDgFrVGM",
  }
}
```

Delete Room Response

```
HTTP/1.1 200 OK
Server: nginx/1.10.3
Date: Mon, 15 Mar 2021 15:32:37 GMT
Content-Type: application/json; charset=utf-8
{
  "roomId" : "1gf2SgxX4vdhDg"
}
```