Сервер обработки сообщений E-message

Оглавление

[Бизнес-аналитика 3](#_Toc110079457)

[Границы проекта 3](#_Toc110079458)

[Перечень пользователей и заинтересованных лиц 4](#_Toc110079459)

[Глоссарий 4](#_Toc110079460)

[Система из предметной области 4](#_Toc110079461)

[1. Бизнес-процессы 4](#_Toc110079462)

[2. Модель предметной области 4](#_Toc110079463)

[Системная аналитика 6](#_Toc110079464)

[Прецеденты и сценарии 6](#_Toc110079465)

[Требования проекта 6](#_Toc110079466)

[1. Бизнес-правила 6](#_Toc110079467)

[2. Атрибуты качества 7](#_Toc110079468)

[3. Функциональные требования 7](#_Toc110079469)

[Архитектура 8](#_Toc110079470)

[Интерфейс 9](#_Toc110079471)

[Проектирование 10](#_Toc110079472)

[1. Разработка алгоритмов 10](#_Toc110079473)

[2. Кодирование и отладка 10](#_Toc110079474)

# Бизнес-аналитика

## Границы проекта

/\*Указываем что мы делаем и что мы не делаем. То есть, очерчиваем зону ответственности проекта.\*/

## Перечень пользователей и заинтересованных лиц

## Глоссарий

## Система из предметной области

### Бизнес-процессы

### Модель предметной области

#### Диаграмма классов

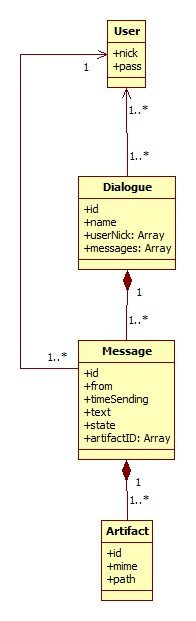


Рисунок — диаграмма классов приложения

#### Диаграмма состояний

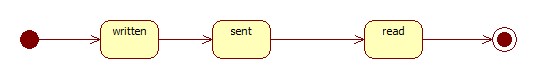


Рисунок — диаграмма состояний сообщения

# Системная аналитика

## Прецеденты и сценарии

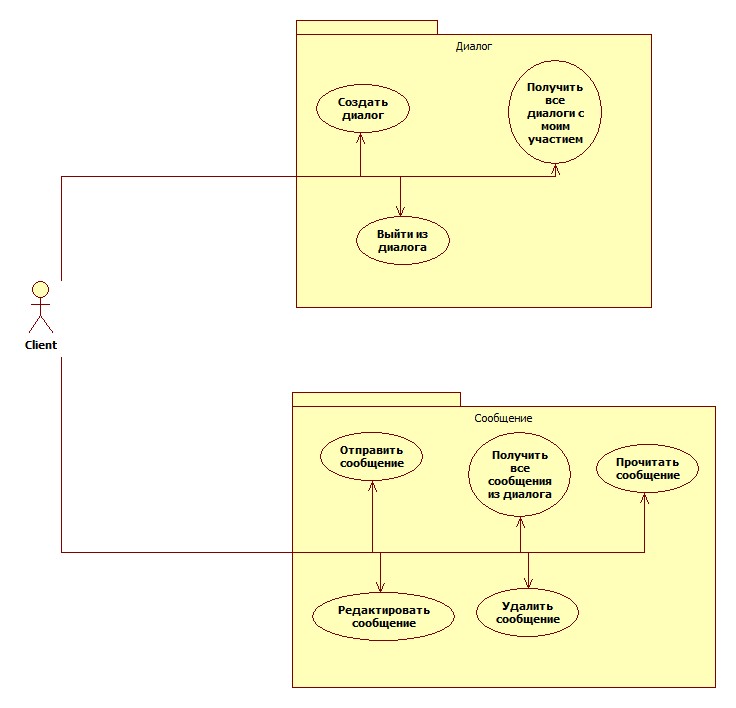


Рисунок — диаграмма прецедентов приложения

## Требования проекта

/\*Взгляд на систему изнутри, более чёткое описание модели, полученной в бизнес-анализе, и описание архитектурных решений, необходимых для её реализации\*/

### Бизнес-правила

/\*Перечень условий, ограничений, правил, проистекающих непосредственно из предметной области\*/

### Атрибуты качества

/\*Перечень качественных характеристик системы, которые необходимо реализовать и которые по возможности должны быть измерены в количественных мерах\*/

**Атрибуты.Производительность.Многопоточность.** Приложение должно эффективно распараллеливать производимые процессы.

**Атрибуты.Производительность.Доступ к модели.** Приложение должно кэшировать состояние модели в in memory database.

### Функциональные требования

/\*Свойства и условия, которыми должны обладать конкретные функции системы\*/

**Функции.Обслуживание Соединений.** Приложение должно слушать указанный в конфигурации порт, при этом иметь множество процессов (на том же или других узлах), обслуживающих запросы клиентов.

**Функция.Процессы-обработчики.** Количество процессов, обрабатывающих входящие запросы задаётся перед запуском сервера из конфигурационного файла.

**Диалог.Удаление.** Клиенты имеют право только выйти из диалога. Если в диалоге не осталось участников, то он автоматически удаляется со всеми содержащимися сообщениями.

**Соообщение.Удаление.** При удалении сообщения автоматически уничтожаются его артефакты при наличии.

## Архитектура

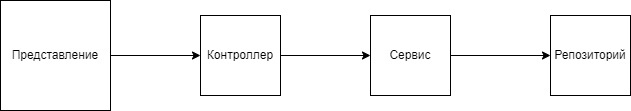


Рисунок — архитектура кода серверного приложения

Приложение состоит из слоёв:

* Представление — слой, отвечающий за общение с клиентами, сериализацию сущностей модели в человекочитаемое представление;
* Контроллер — предоставляет интерфейс воздействия на модель слою представления. Отвечает за сложную логику обработки сущностей — каждая функция может вызвать одну или несколько команд слоя сервисов как для декларированной, так и для других сущностей.
* Сервис — модули данного слоя предоставляют перечень атомарных и не связанных друг с другом функций, направленных на изменение состояния одной декларированной сущности;
* Репозиторий — функции данного слоя занимаются чтением/записью сущностей модели в постоянную память и кэш.

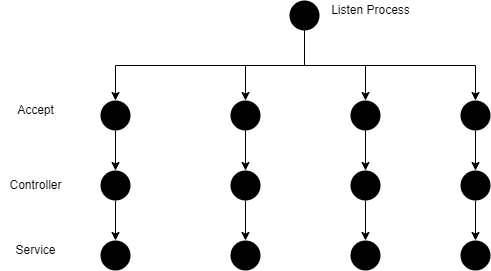


Рисунок — граф управляющих и информационных связей приложения

Процесс, слушающий сокет, выполняет инициализацию приложения, создаёт N принимающих процессов, где N – число, прочитанное из конфигурационного файла.

Каждый принимающий процесс занимается обслуживанием клиентских запросов —вызов функции контроллера, соответствующей поступившему сообщению.

## Интерфейс

Интерфейс приложения построен на передаче от клиента по TCP-соединению списка, содержащего:

1. Название действия, например, прочитать сообщение, отправить сообщение и т.д;
2. Перечень входных аргументов: объекты, идентификаторы, содержимое полей.

Информация пересылается в текстовом виде.

Сервер, получив запрос и выполнив его обработку присылает пользователю один из ответов:

1. ok;
2. error, reasonMsg.

# Проектирование

### Разработка алгоритмов

#### Генерация принимающих процессов

### Кодирование и отладка

#### Слой репозитория

Все поисковые методы на уровне репозитория могут вернуть: пустой список, список элементов (или элемент, если подразумевается единственный результат), кортеж-ошибку.

Репозиторий – абстракция для получения данных. Слой предоставляет интерфейс получения записей. Каждый метод интерфейса должен использоваться внутри транзакций.