# Практическая работа. Лекция 9. Декомпозиция на функциональные компоненты

## Пользовательские сценарии

Сценарий 1

Есть клиент банка

И есть банк

Когда клиент совершает покупку,

тогда банк хочет уведомить клиента о результатах выполненного платежа,

Получив результат выполнения операции, формируется информационное сообщение,

сообщение отправляется в смс-сервис,

Тогда клиент банка получает смс-сообщение на телефон

Сценарий 2

Есть отправленное смс-сообщение банком,

И есть клиент и банк

Когда у банка возникает желание получить статус отправленного сообщения,

Тогда формируется сообщение о получении статуса сообщения,

Когда сообщение отправляется в смс-сервис,

Тогда банк получает ответ, в котором указан статус отправленного ранее сообщения

Сценарий 3

Есть администратор системы

И есть новый провайдер по отправке сообщения

Когда администратор получает распоряжение о добавлении нового провайдера по отправке смс-сообщений,

Тогда администратор регистрирует данные провайдера в смс-сервисе;

Сценарий 4

Есть служба поддержки и система мониторинга

И есть смс-сообщения, зафиксированные в смс-сервисе,

Служба поддержки заходит в систему мониторинга смс-сервиса

и следит за статусами отправленных сообщений

Когда служба поддержки фиксирует превышение определенного лимита количества сообщений со статусом “неотправленные”,

Тогда служба поддержки временно приостанавливает отправку сообщений данному провайдеру

Сценарий 5

Есть служба поддержки

И есть провайдер, которому приостановлена отправка сообщений

Когда служба поддержки связывается с провайдером и положительно решает вопрос о дальнейшем взаимодействия сервисов,

Тогда служба поддержки вновь открывает отправку сообщений данному провайдеру

Сценарий 6

Есть банк,

Есть клиенты банка,

есть несколько шаблонов сообщений, которые соответствуют разным бизнес-контекстам

Когда банк решает создать несколько шаблонов смс-уведомлений в зависимости от бизнес-контекста,

Тогда он входит в личный кабинет смс-сервиса и создает шаблоны под бизнес-контекст

Сценарий 7

Есть бизнес

Существует потребность получать статистические данные по количеству отправленных сообщений за определенный период, чтобы полученные данные преобразовать в деньги- доход,

Когда представитель бизнеса делает запрос на получение отчета отправленных сообщений за определенный период,

Тогда сервис формирует отчет и отправляет его в качестве ответа пользователю.

## Модель предметной области

### Решение 1(по сущностям)

Сущности(Рис. 1):

1. Message – сообщение:
   1. Id – идентификатор сообщения;
   2. Text – текст сообщения
   3. Addressto – адрес получателя(телефон)
   4. From – от кого
   5. Asynch – срочное сообщение или нет
   6. Tmpl – имя шаблона
   7. Extid – идентификатор сообщения провайдера
   8. Status - статус
2. Msgtemplate – шаблон:
   1. Name - название
   2. Description - описание
   3. Value - шаблон
   4. Enabled – включен или отключен
3. Client - клиент
   1. Name - имя
   2. Surname - фамилия
   3. Email -почта
   4. Enabled - включен или отключен
4. Msgreport - отчет
   1. Quantity – количество сообщений
   2. Status – статус сообщений(группировка по статусу)
   3. Client - клиент
5. Provider – внешние провайдеры
   1. Name - название
   2. url – URL адрес внешнего провайдера
   3. parameter – параметры взаимодействия
   4. protocol - протокол
   5. enabled- включен или отключен
6. Credential – данные для аутентификации клиентов
   1. Login - логин
   2. Password - пароль
7. Extcredential – полномочия для подключения к внешним провайдерам
   1. Login - логин
   2. Password - пароль
8. User – внутренние пользовтаели
   1. Name - имя
   2. Surname - фамилия
   3. Email - почта
   4. Role - роль
   5. Enabled - включен или отключен
9. Usercredential – данные для аутентификации внутренних пользователей
   1. Login - логин
   2. Password - пароль
10. Role – роли: администратор, служба поддержки, аналитики и т.п
    1. Name – название роли
    2. Enabled - включен или отключен
11. Queuemsg – распределение сообщений
    1. Message - Сообщение
    2. Name - Название очереди

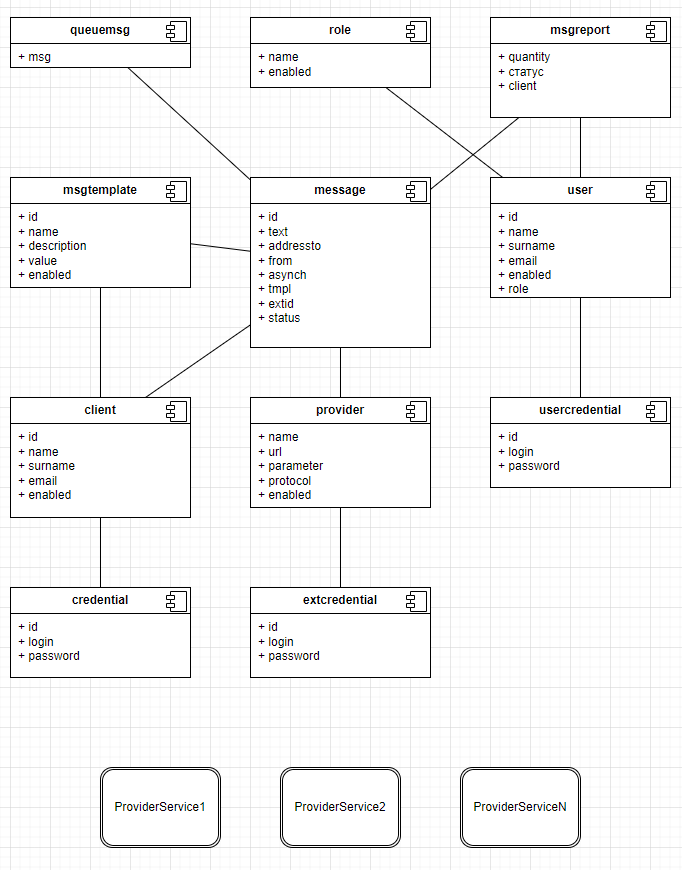


Рис. - Сущности объектной модели

### Решение 2 (по бизнес-функциям)

Сценарий 1 состоит из следующих процессов(Рис. 2):

1. Отправить уведомление;
2. Авторизация запроса;
3. Проверка сообщения;
4. Определение шаблона;
5. Определение провайдера;
6. Отправка сообщения провайдеру.



Рис. - Схема бизнес-процесса по отправке сообщения

Сценарий 2 состоит из следующих процессов(Рис. 3):

1. Получить статус сообщения;
2. Авторизация запроса;
3. Определение провайдера;
4. Отправка сообщения провайдеру.



Рис. - Схема бизнес-процесса по получению статуса сообщений

Сценарий 3 состоит из следующих процессов(Рис. 4):

1. Заключение соглашения с новым провайдером;
2. Вход администратора;
3. Создание записи о новом провайдере.



Рис. - Схема бизнес-процесса по регитрации нового провайдера

Сценарии 4 и 5 состоят из следующих процессов(Рис. 5):

1. Получение статистической информации;
2. Фиксирование порога не отправленных сообщений по провайдерам;
3. Отправка уведомления о событии;
4. Вход в смс-сервис представителя службы поддержки;
5. Найти провайдера;
6. Блокирование провайдера;
7. Связаться с провайдером;
8. Если проблема решена, то разблокировать провайдера;
9. Если нет, то передать уведомление менеджеру о блокировке провайдера



Рис. - Схема бизнес-процесса по блокированию и разблокированию провайдера

Сценарий 6 состоит из следующих процессов(Рис. 6):

1. Банк сформировал шаблон по отправке поздравлений “С Новым Годом” своим клиентам;
2. Клиент входит в личный кабинет смс-сервиса;
3. Клиент создает шаблон по отправке поздравлений “С Новым Годом”.



Рис. - Схема бизнес-процесса по созданию шаблона

Сценарий 7 состоит из следующих процессов:

1. Появился запрос на подсчет дохода за неделю по сервису;
2. Пользователь входит в систему;
3. отправляет запрос на получение отчета о количестве отправленных сообщений за неделю по всем провайдерам;
4. получает отчет и по его данным считает доход.

На основе бизнес-процессов можно выделить следующие контексты(Рис. 7, Рис. 8, Рис. 9):

1. Контекст отчет – отчеты;
2. Контекст сообщение – регистрация и контроль всех поступивших сообщений;
3. Контекст клиент – данные для аутентификации и авторизации клиентов сервиса;
4. Контекст провайдер;
5. Контекст пользователь – внутренние пользователи.
6. Контекст очередь – отправка сообщений из очереди.

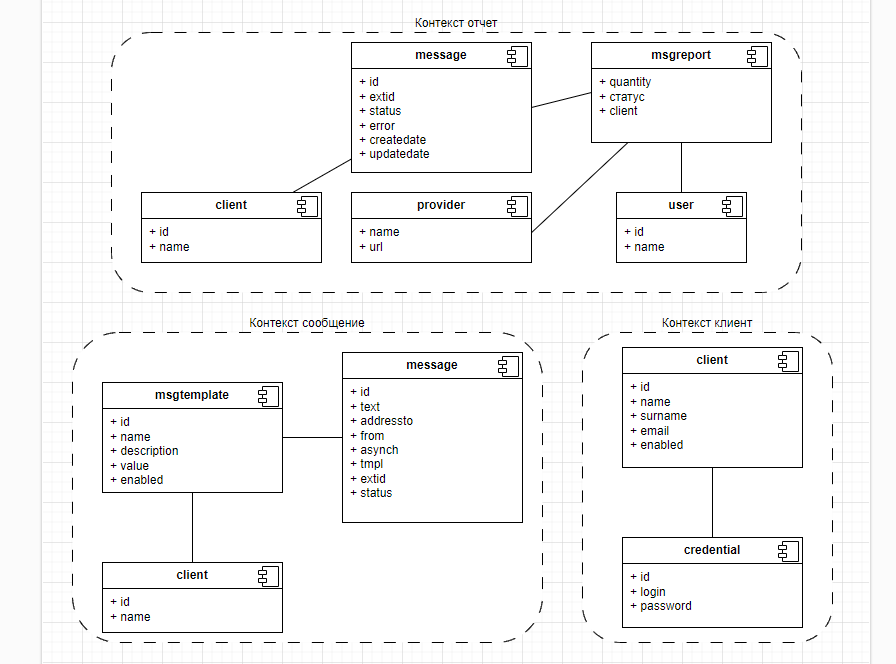


Рис. - Контексты отчет, сообщение, клиент

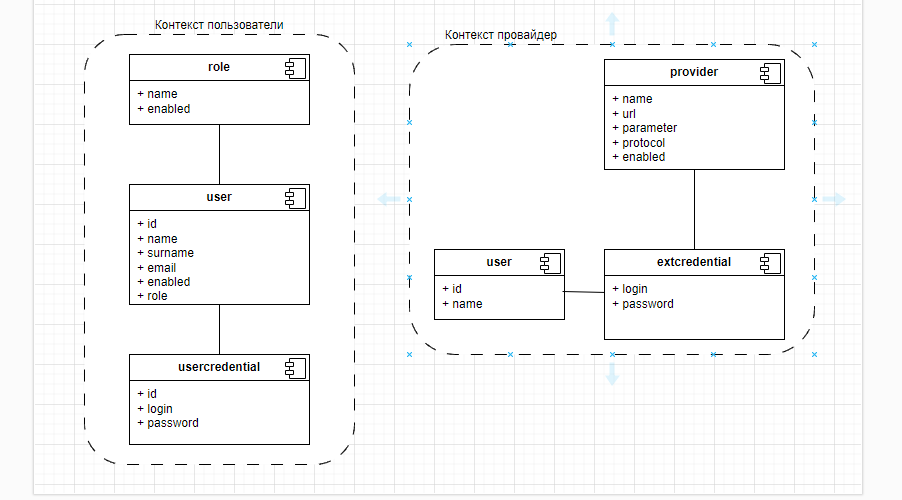


Рис. – Контексты пользователи, провайдер

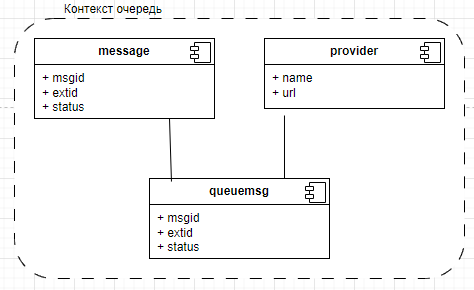


Рис. - Контекст очередь

## Модифицируемость

Сценарий изменений:

Клиентами могут быть внешние системы, так и физические лица;

* Провайдер может поменять параметры для взаимодействия - необходимо не только создавать и блокировать провайдера, но и изменять его данные;
* Создание новых отчетов, сформированных на основе новых данных;
* Возможность группировки отправки сообщений по имени шаблона. Сейчас каждому сообщению м.б. указан шаблон;
* Клиент хочет получить доступ к своим личным данным в сервисе и иметь возможность менять их;
* Могут быть выставлены приоритеты отправки сообщений по типу или клиенту - у каждого могут свои SLA;
* В сообщении могут появиться новые поля для определения приоритета и типа сообщения;
* Поиск сообщений с определенным статусом для выявления ошибок.

Оценка стоимости изменений по решению 1:

* Провайдер может поменять параметры для взаимодействия - необходимо не только создавать и блокировать провайдера, но и изменять его данные.

C(provider)+C(user)

* Создание новых отчетов, сформированных на основе новых данных.

C(msgreport)+C(message)

* Возможность группировки отправки сообщений по имени шаблона. Сейчас каждому сообщению м.б. указан шаблон.

C(message)+C(msgtemplate)+C(provider)

* Клиент хочет получить доступ к своим личным данным в сервисе и иметь возможность менять их.

C(client)+C(новый объект)

* Могут быть выставлены приоритеты отправки сообщений по типу или клиенту - у каждого могут свои SLA.

C(message)+C(queuemsg)

* В сообщении могут появиться новые поля для определения приоритета и типа сообщения.

C(message)+C(provider)

* Поиск сообщений с определенным статусом для выявления ошибок.

C(message)+C(msgreport)

Оценка стоимости изменений по решению 2:

* Провайдер может поменять параметры для взаимодействия - необходимо не только создавать и блокировать провайдера, но и изменять его данные.

C(контекст провайдер)

* Создание новых отчетов, сформированных на основе новых данных.

C(контекст отчет)

* Возможность группировки отправки сообщений по имени шаблона. Сейчас каждому сообщению м.б. указан шаблон.

C(контекст сообщение)

* Клиент хочет получить доступ к своим личным данным в сервисе и иметь возможность менять их.

C(контекст клиент)

* Могут быть выставлены приоритеты отправки сообщений по типу или клиенту - у каждого могут свои SLA.

C(контекст сообщения)

* В сообщении могут появиться новые поля для определения приоритета и типа сообщения.

C(контекст сообщения)+С(контекст очередь)

* Поиск сообщений с определенным статусом для выявления ошибок.

C(контекст очередь)

|  |  |  |  |
| --- | --- | --- | --- |
| № | Характеристика | Решение 1 | Решение 2 |
| 1 | Архитектурный стиль | Модульный монолит,  микрокернел монолит | От мололита до микросервисов |
| 2 | Гибкость изменений | До тех пор, пока не потребуется включать много логики, Структура может быть сильно связанной | Изменения будут касаться в основном только тех объектов, которые находятся внутри ограниченного контекста |
| 3 | Стоимость | В долгосрочной перспективе изменения могут стоить дороже | Изменения дешевле |
| 4 | Масштабируемость | Сложно, вертикальное масштабирование | Легко масштабируется, поддержка горизонтального масштабирования |
| 5 | Ответственность | Ответственность не только за один модуль, но и за все приложение в целом | Ответственность только за ту часть приложения, которая ограничена контекстом. За каждый контекст может отвечать своя команда |

В целом второе решение в долгосрочной перспективе имеет ряд очевидных преимуществ, в основном это вызвано тем, что проводится четкая граница между отдельными контекстами. Первое решение следует применять в стартапах, либо в небольших проектах – они требует гораздо меньше затрат на поддержку, разработку. Второе решение позволяет легче вводить изменения в ПО, часто не затрагивая другие контексты(сервисы, микросервисы, модули), но возможно потребует больше ресурсов на поддержку инфраструктуры. Еще одним достоинством второго решения является быстрое включение в работу нового сотрудника команды – ему на первых порах достаточно знать, как работает одна часть приложения, ограниченная одним контекстом.

В итоге выбор решения во многом зависит от того, какие задачи ставит перед собой бизнес, команда.