Магистерская диссертация

Интеграция системы дистанционного банковского обслуживания и платежной системы «Рапида»

Ядгарова Юлия Владимировна

Научный руководитель: старший преподаватель каф. РК-9

Урусов Андрей Витальевич

Москва, 2013

Актуальность работы

- Платежи от физических лиц являются объектом особого внимания со стороны банковского розничного бизнеса.
- Предоставление физическим лицам удобного и защищенного инструмента для проведения оплат необходимых товаров и услуг безналичным способом в онлайн режиме.
- Разработка типовых решений, позволяющие решать задачи интеграции информационных систем различного типа.

Цель работы:

- Исследование архитектур распределенных систем применительно к области банковской автоматизации и разработка модуля для интеграции системы дистанционного банковского обслуживания и платежного сервиса.

* **Система дистанционного банковского обслуживания (СДБО)** – система для предоставления банковских услуг на основании распоряжений, передаваемых клиентом удаленным образом (то есть без его визита в банк), чаще всего с использованием компьютерных и телефонных сетей

* **Платежная система -** совокупность правил, процедур и технической информации, обеспечивающих перевод стоимости от одного субъекта экономики другому.

Методы исследования:

Анализ проблемной области

Классификация существующих подходов к интеграции распределенных ИС.

Научная новизна:

1. Проанализированы различные подходы к построению распределенных систем и выявлены достоинства и недостатки существующих архитектур применительно к проблемной области.
2. В разработанном модуле соединяются 2 подхода к построению распределенных систем.

Постановка задачи

- Обеспечить возможность через интерфейс системы дистанционного банковского обслуживания проводить авансовые платежи через ПС «Рапида»

Основные методы интеграции ИС*:

- Обмен файлами (общие данные).
- Общая база данных (БД), в которой сохраняется общая информация.
- Удаленный вызов процедур(Для распределенных систем – системы обмена сообщениями):

  - Компонентная объектная модель Component Object Model (COM).

  - Архитектура брокеров объектных запросов Common Object Request Broker Architecture (CORBA)

  - Java RPC (Remote Procedure Call).

  - Веб-сервисы, в том числе, ESB и SOAP.

- Асинхронный обмен сообщениями:

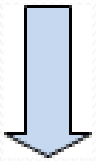
  - Microsoft Message Queue (MSMQ),

  - Java Message Service (JMS).

*(Хоп Г., Вульф Б.. Шаблоны интеграции корпоративных приложений. – М.: Изд-во «Вильяме», 2007.– 672 с.)

Особенности автоматизации банковских систем

- Слабосвязанная архитектура модулей для отказоустойчивости каждого модуля
- Наличие средств защиты информации
- Возможность работы в реальном времени или с высокой частотой синхронизации процессов
- Возможность легкой адаптации к изменяющимся условиям законодательства
- Наличие дополнительных легко встраиваемых функциональных возможностей
- Обеспечение транзакционности всех процессов



Концепция сервисно-ориентированной архитектуры (SOA) - основа для стратегии современного банка в области интеграции

Основные методы интеграции ИС*:

- Обмен файлами (общие данные).
- Общая база данных (БД), в которой сохраняется общая информация.
- **Удаленный вызов процедур(Для распределенных систем – системы обмена сообщениями):**

  - Компонентная объектная модель Component Object Model (COM).

  - Архитектура брокеров объектных запросов Common Object Request Broker Architecture (CORBA)

  - Java RPC (Remote Procedure Call).

  - Веб-сервисы, в том числе, ESB и SOAP.

- **Асинхронный обмен сообщениями:**

  - Microsoft Message Queue (MSMQ),

  - Java Message Service (JMS).

*(Хоп Г., Вульф Б.. Шаблоны интеграции корпоративных приложений. – М.: Изд-во «Вильяме», 2007.– 672 с.)

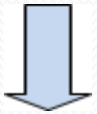
ООО НКО «Рапида»

Крупная сеть пунктов приема платежей, включающая в себя кассы магазинов бытовой техники, сетей сотовой связи, терминалы самообслуживания и банкоматы.

- Вызов процедур: https://gate.rapida.ru/gate.
- Передача параметров: метод GET согласно RFC 1945, RFC 2616, RFC 3986.
- Кодировка запросов: windows-1251 в формате «URL-encoded».
- Информационное взаимодействие Участника Системы и ПС «Рапида» осуществляется с использованием протокола безопасной передачи данных HTTPS (HTTP over SSL), при этом, аутентификация\авторизация Участника Системы происходит путем предъявления цифрового сертификата формата X.509 v3.

Основные типы запросов к ПС «Рапида»

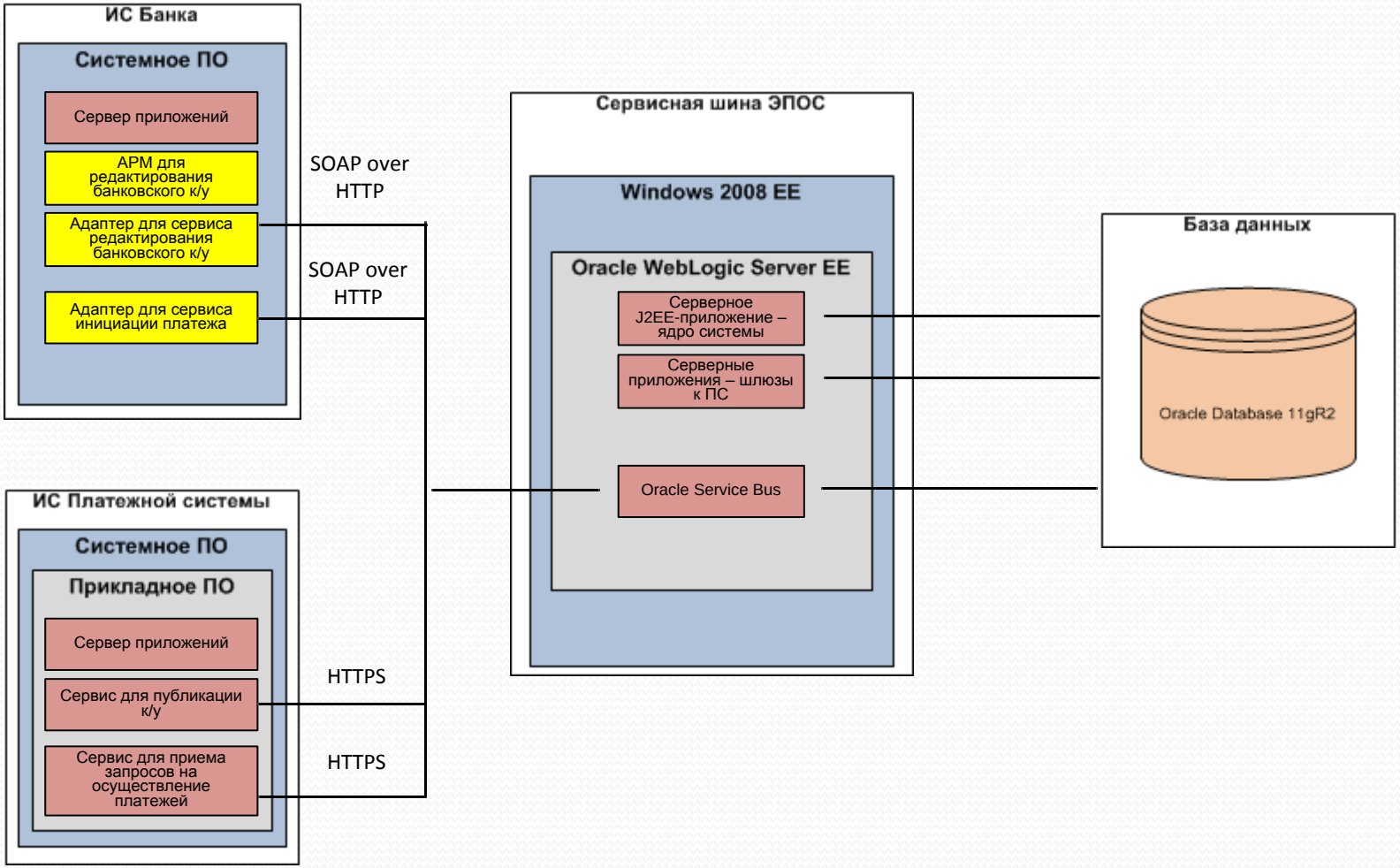
- Запрос на выгрузку справочника доступных услуг ***getFee()*** - **синхронный**;
- Запрос проверки правильности реквизитов платежа ***checkPayment()*** **– синхронный**;
- Запрос на проведение платежа – ***pay()*** **асинхронный**;
- Запрос проверки статуса платежа – ***checkPay()*** - **асинхронный;**
- Запрос на отмену платежа – ***cancel()*** – **синхронный**



Выбранные методы интеграции:

- **Веб-сервисы (JAX-WS + SOAP + ESB)** – для взаимодействия с ДБО
- **Java RPC + JMS (очереди сообщений)** – для взаимодействия с ПС

Логическая архитектура ИС системы «Эпос»



Физическая архитектура ИС «ЭПОС»

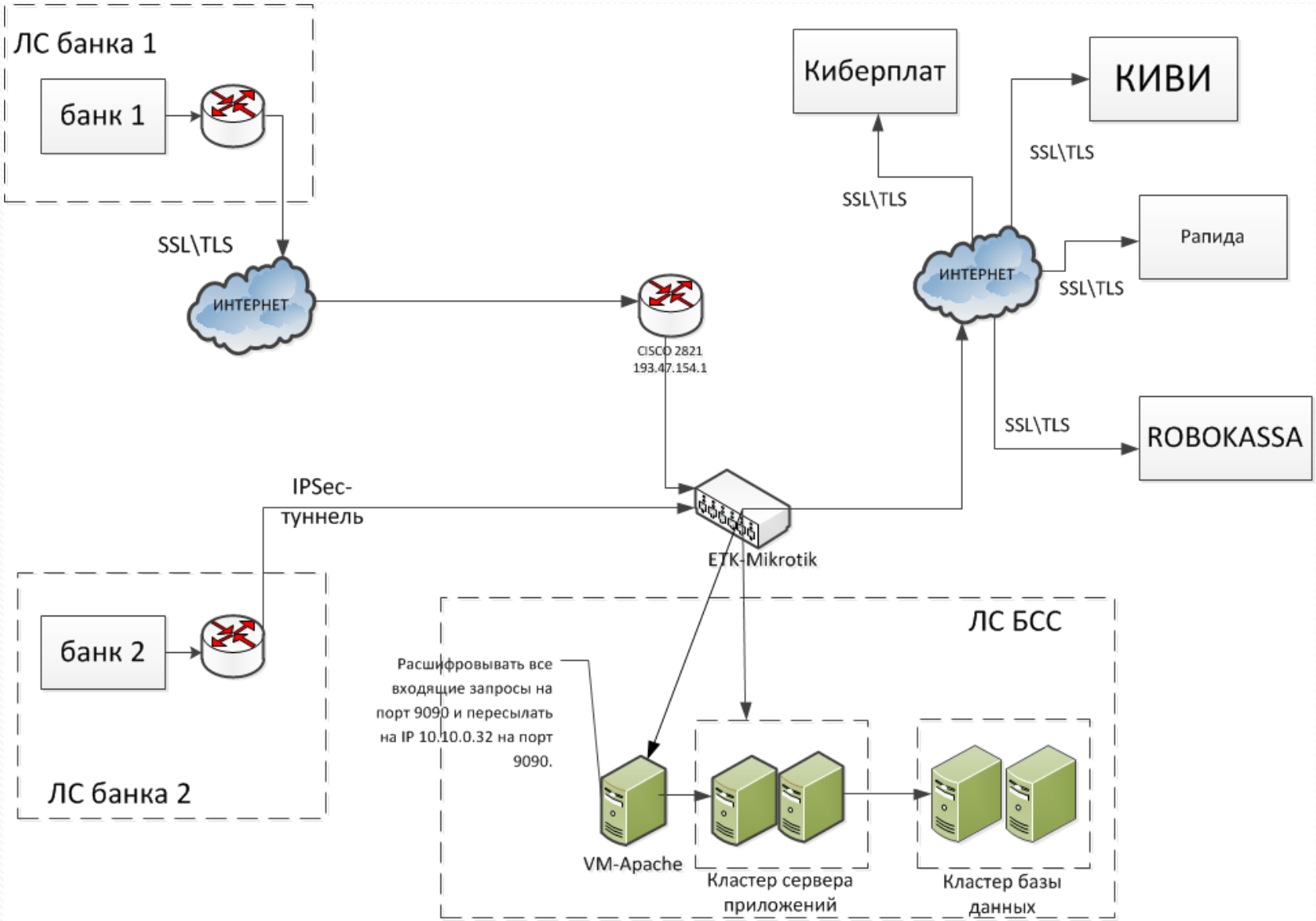
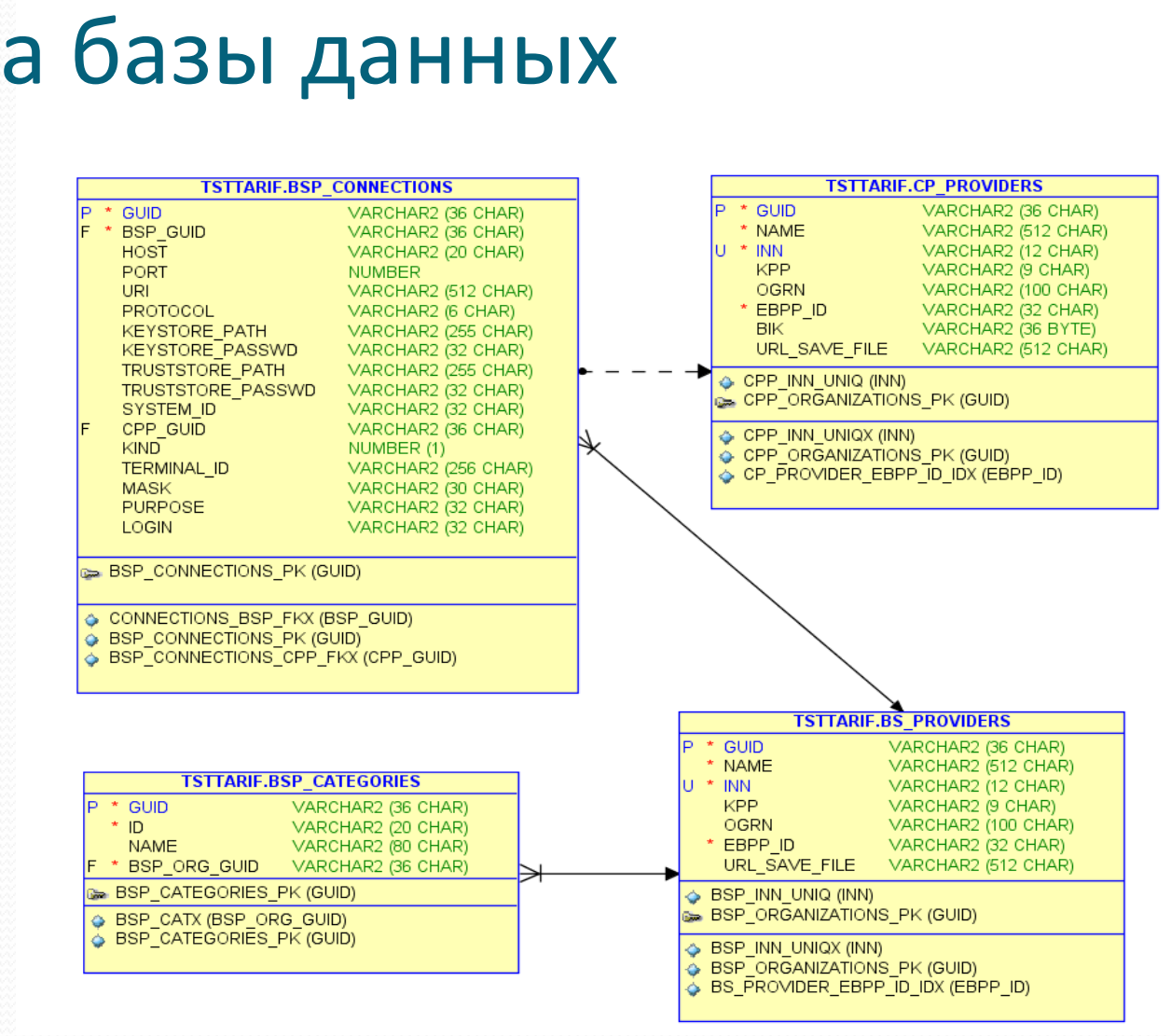
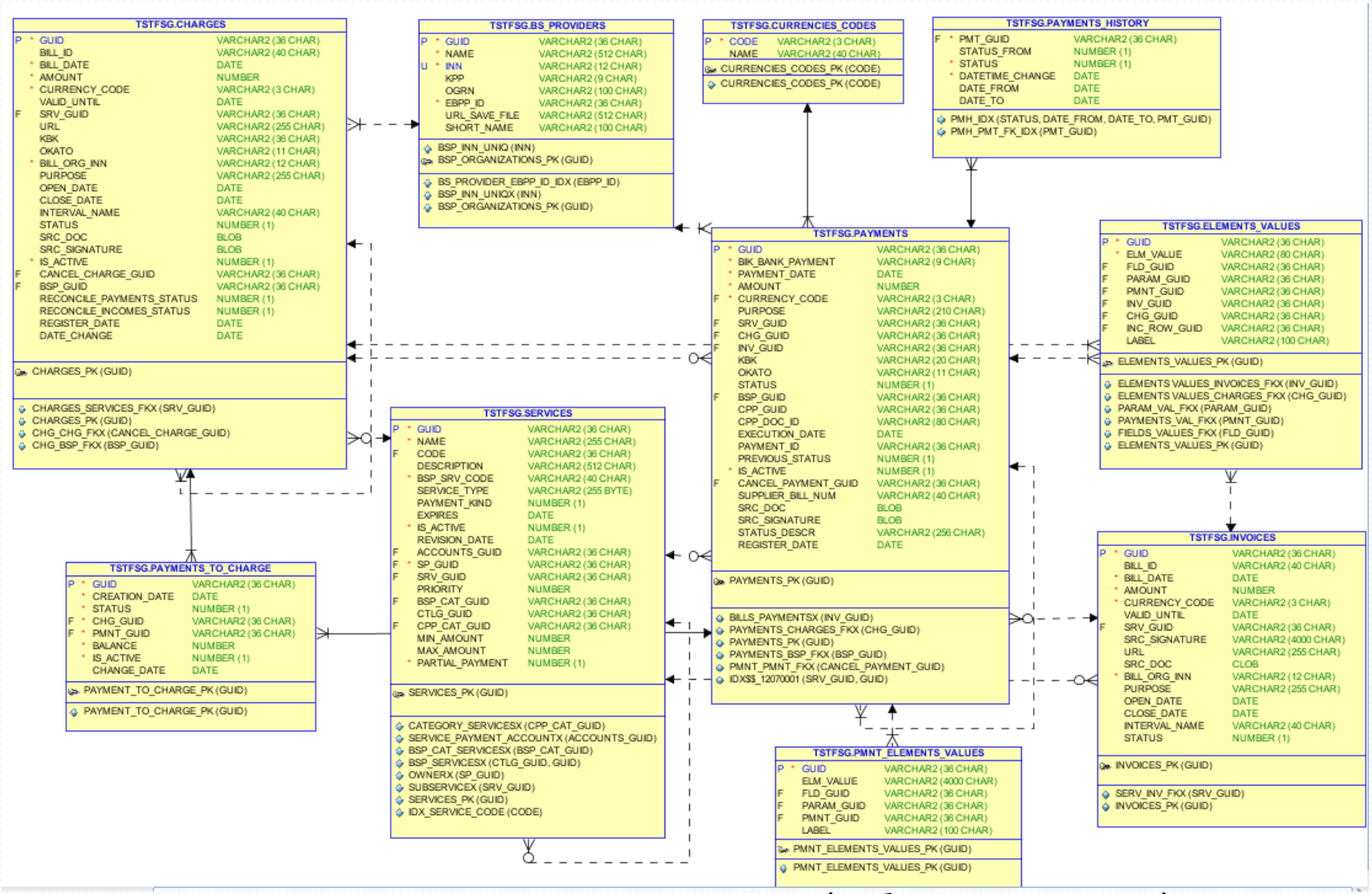


Схема базы данных



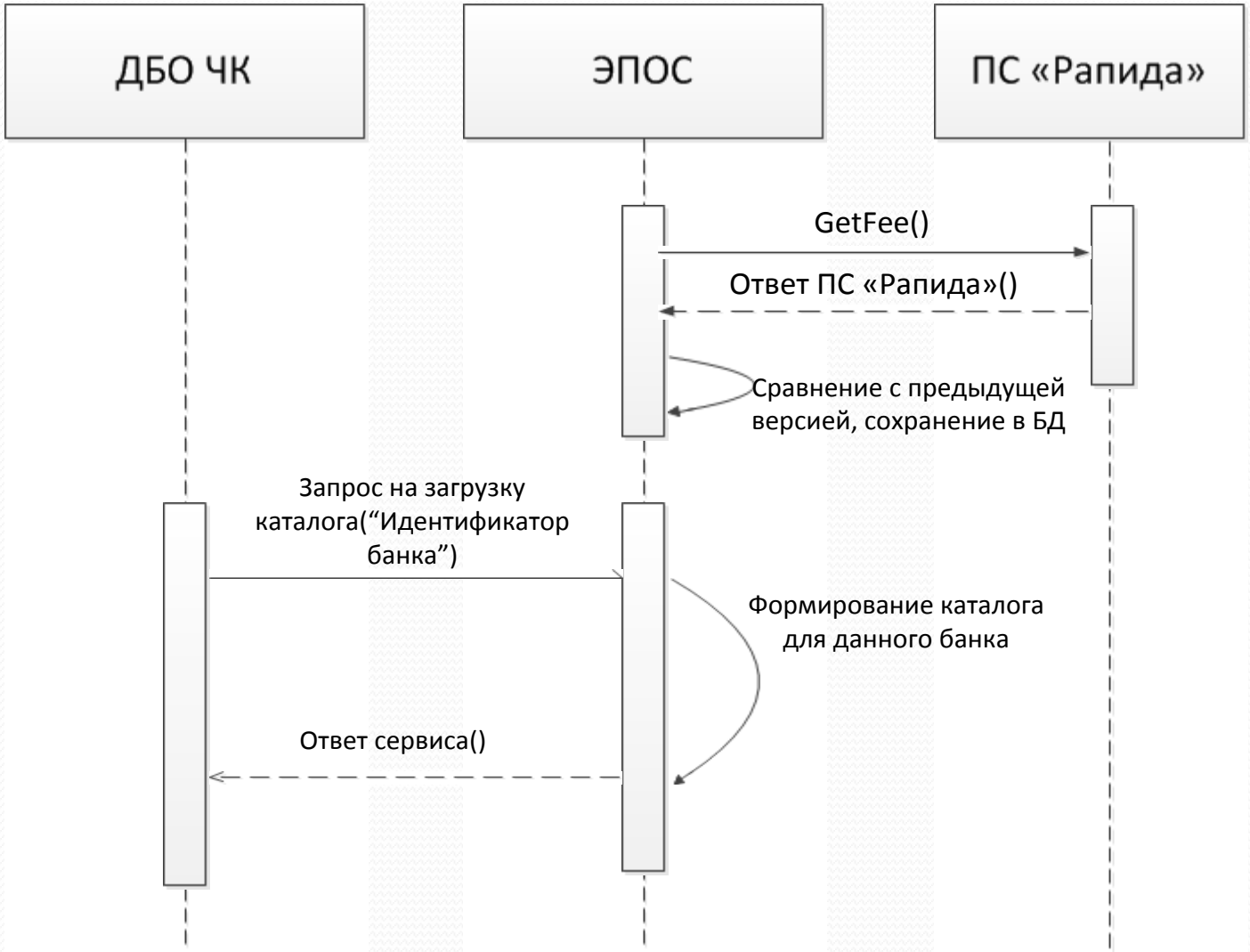
Подключение участников системы (таблица bsp_connections)

Схема базы данных

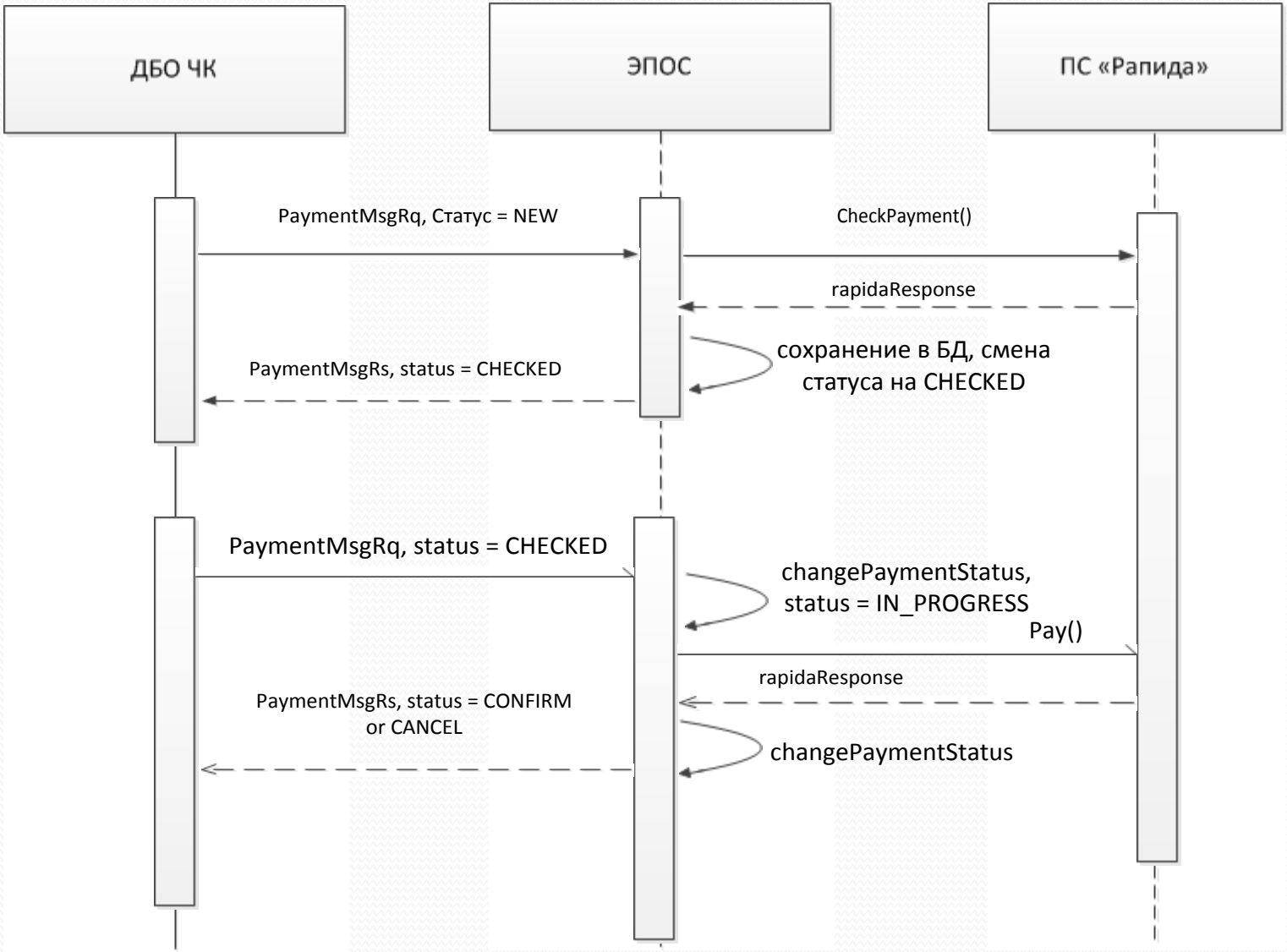


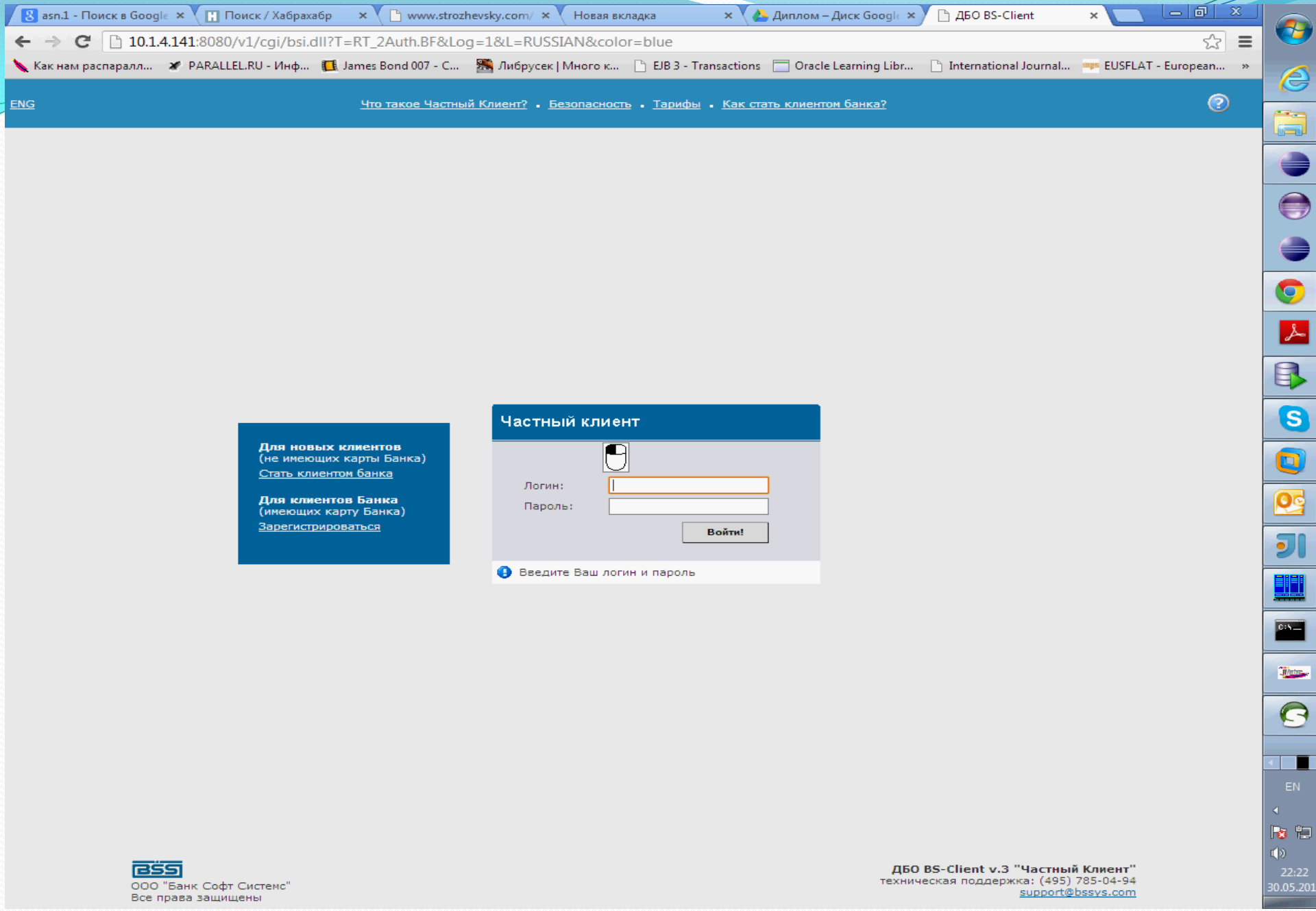
Платежи через систему(таблица payments)

Запрос каталога услуг

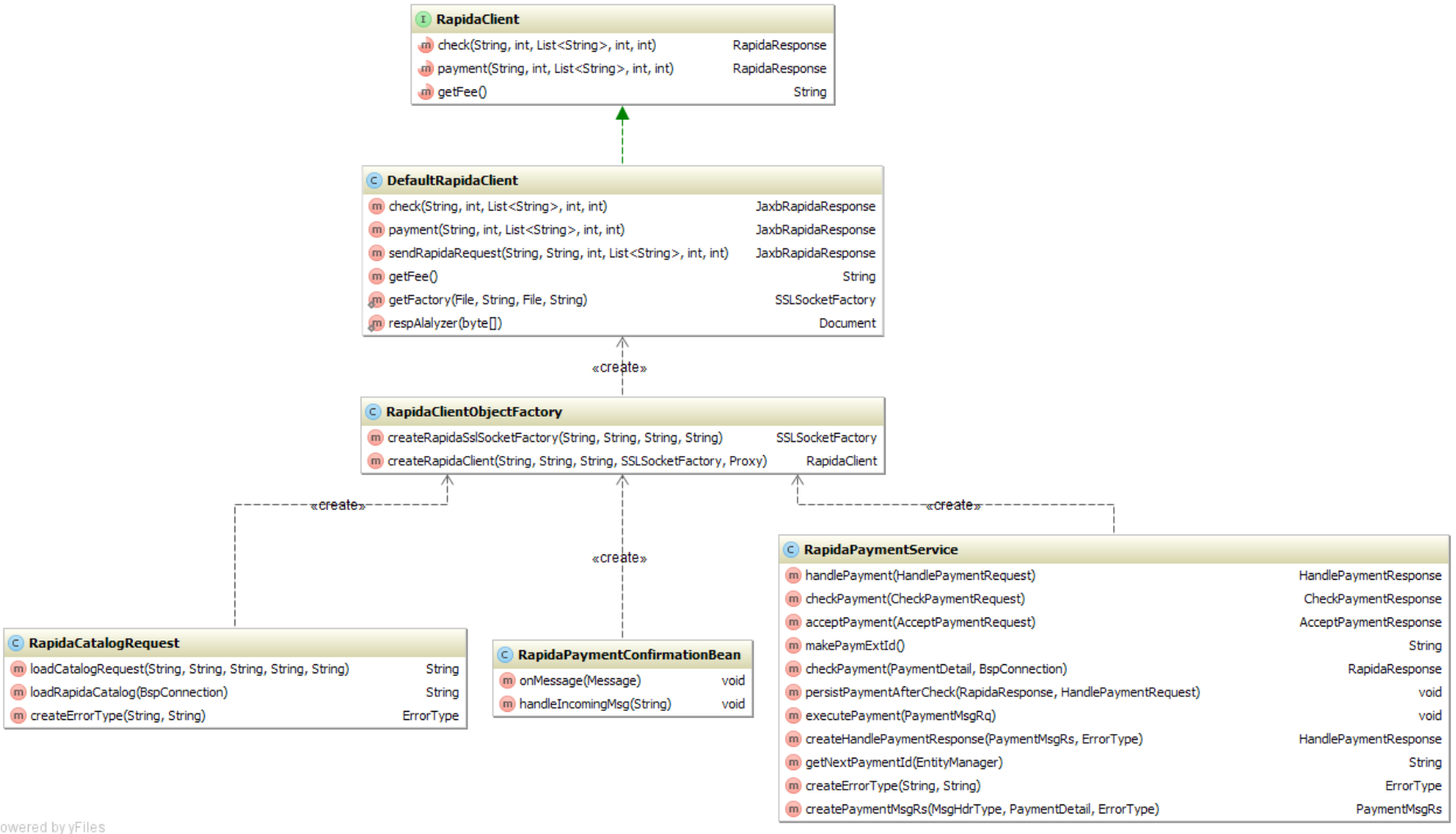


Запрос на платеж

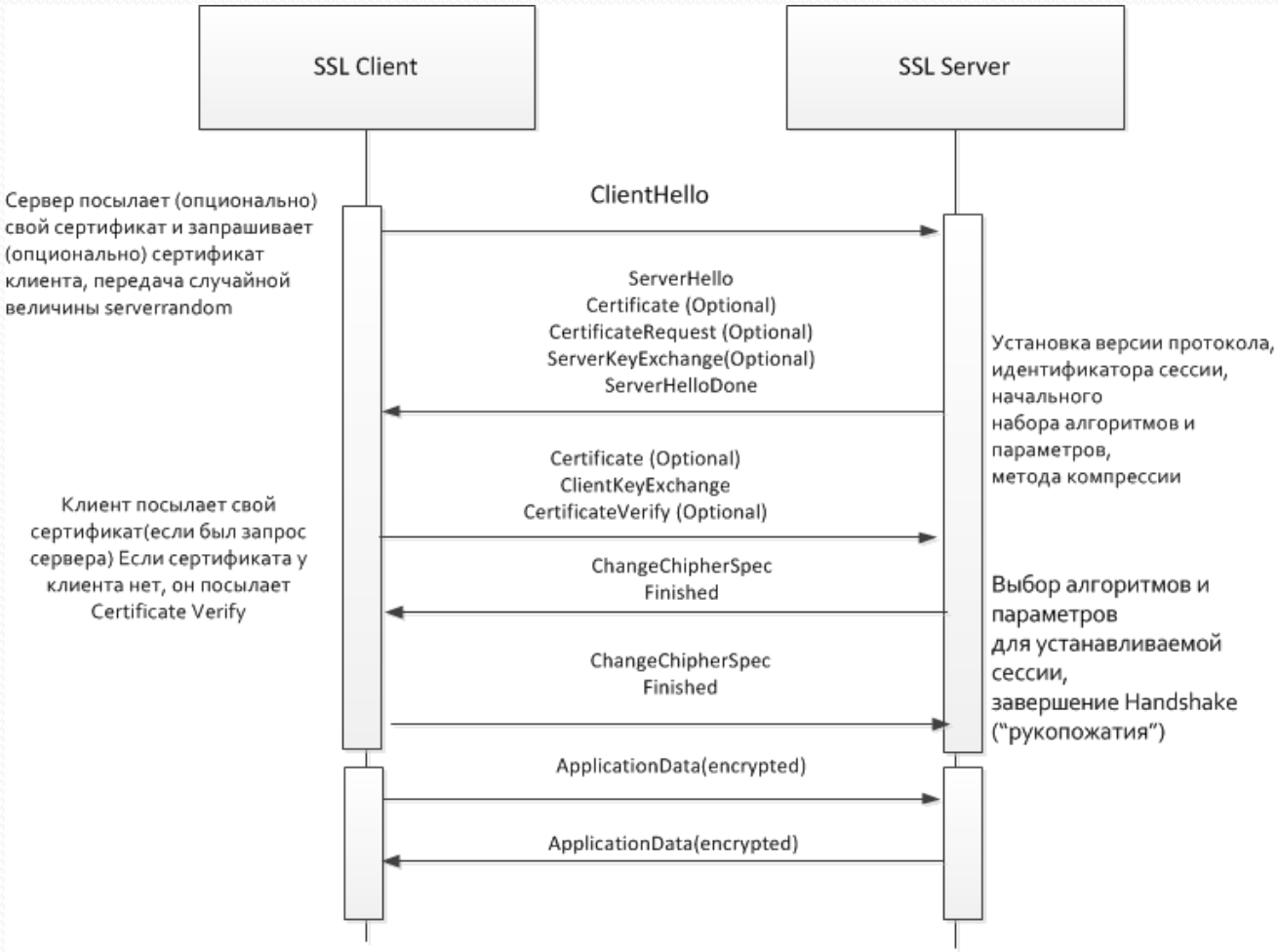



asn.1 - Поиск в Google
Поиск / Хабрахабр
www.strozhevsky.com/
Новая вкладка
Диплом – Диск Google
ДБО BS-Client
10.1.4.141:8080/v1/cgi/bsi.dll?T=RT_2Auth.BF&Log=1&L=RUSSIAN&color=blue
Как нам распаралл...
PARALLEL.RU - Инф...
James Bond 007 - C...
Либрусек | Много к...
EJB 3 - Transactions
Oracle Learning Libr...
International Journal...
EUSFLAT - European...
ENG
Что такое Частный Клиент? . Безопасность . Тарифы . Как стать клиентом банка?
Для новых клиентов
(не имеющих карты Банка)
Стать клиентом банка
Для клиентов Банка
(имеющих карту Банка)
Зарегистрироваться
Частный клиент
Логин:
Пароль:
Войти!
Введите Ваш логин и пароль
ООО "Банк Софт Системс"
Все права защищены
ДБО BS-Client v.3 "Частный Клиент"
техническая поддержка: (495) 785-04-94
support@bssys.com
EN
22:22
30.05.201

UM


RapidaClient
check(String, int, List<String>, int, int) RapidaResponse
payment(String, int, List<String>, int, int) RapidaResponse
getFee() String
DefaultRapidaClient
check(String, int, List<String>, int, int) JaxbRapidaResponse
payment(String, int, List<String>, int, int) JaxbRapidaResponse
sendRapidaRequest(String, String, int, List<String>, int, int) JaxbRapidaResponse
getFee() String
getFactory(File, String, File, String) SSLSocketFactory
respAlalyzer(byte[]) Document
«create»
RapidaClientObjectFactory
createRapidaSslSocketFactory(String, String, String, String) SSLSocketFactory
createRapidaClient(String, String, String, SSLSocketFactory, Proxy) RapidaClient
«create»
«create»
«create»
RapidaCatalogRequest
loadCatalogRequest(String, String, String, String, String) String
loadRapidaCatalog(BspConnection) String
createErrorType(String, String) ErrorType
RapidaPaymentConfirmationBean
onMessage(Message) void
handleIncomingMsg(String) void
RapidaPaymentService
handlePayment(HandlePaymentRequest) HandlePaymentResponse
checkPayment(CheckPaymentRequest) CheckPaymentResponse
acceptPayment(AcceptPaymentRequest) AcceptPaymentResponse
makePaymExtId() String
checkPayment(PaymentDetail, BspConnection) RapidaResponse
persistPaymentAfterCheck(RapidaResponse, HandlePaymentRequest) void
executePayment(PaymentMsgRq) void
createHandlePaymentResponse(PaymentMsgRs, ErrorType) HandlePaymentResponse
getNextPaymentId(EntityManager) String
createErrorType(String, String) ErrorType
createPaymentMsgRs(MsgHdrType, PaymentDetail, ErrorType) PaymentMsgRs
Powered by yFiles

Защита пакетов при передаче данных (использование SSL-соединения)



Генерация сертификата клиента

1. Запрос на сертификат в формате CSR (certificate signing request):

- генерация закрытого ключа для подписи
- создание запроса
- отправка запроса на сервер (информация о владельце ключа + открытый ключ)

2. Получение сертификата, соответствующего закрытому ключу.

3. Создание хранилища сертификатов в формате pkcs#12 для использования в коде.

Запрос на сертификацию:

openssl req -new -newkey rsa:2048 -keyout rsa_key.pem --out certreq.pem

Создание хранилища:

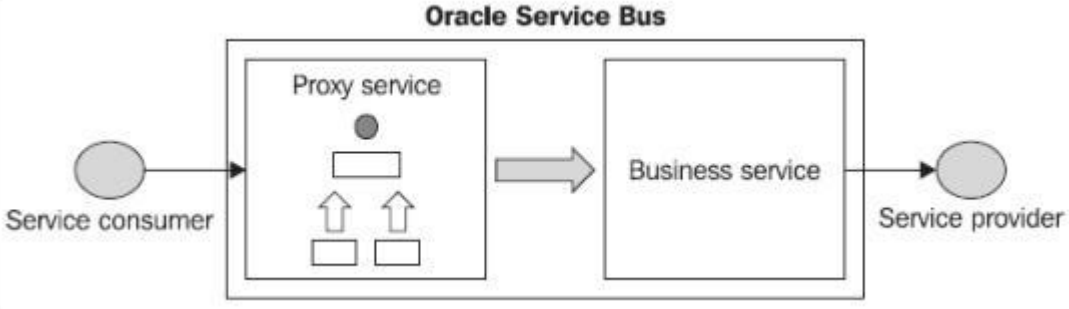
openssl pkcs12 -export -in client.pem -inkey client-key.pem -out client.p12 -name ClientCert

Сервис-шина Oracle Service Bus

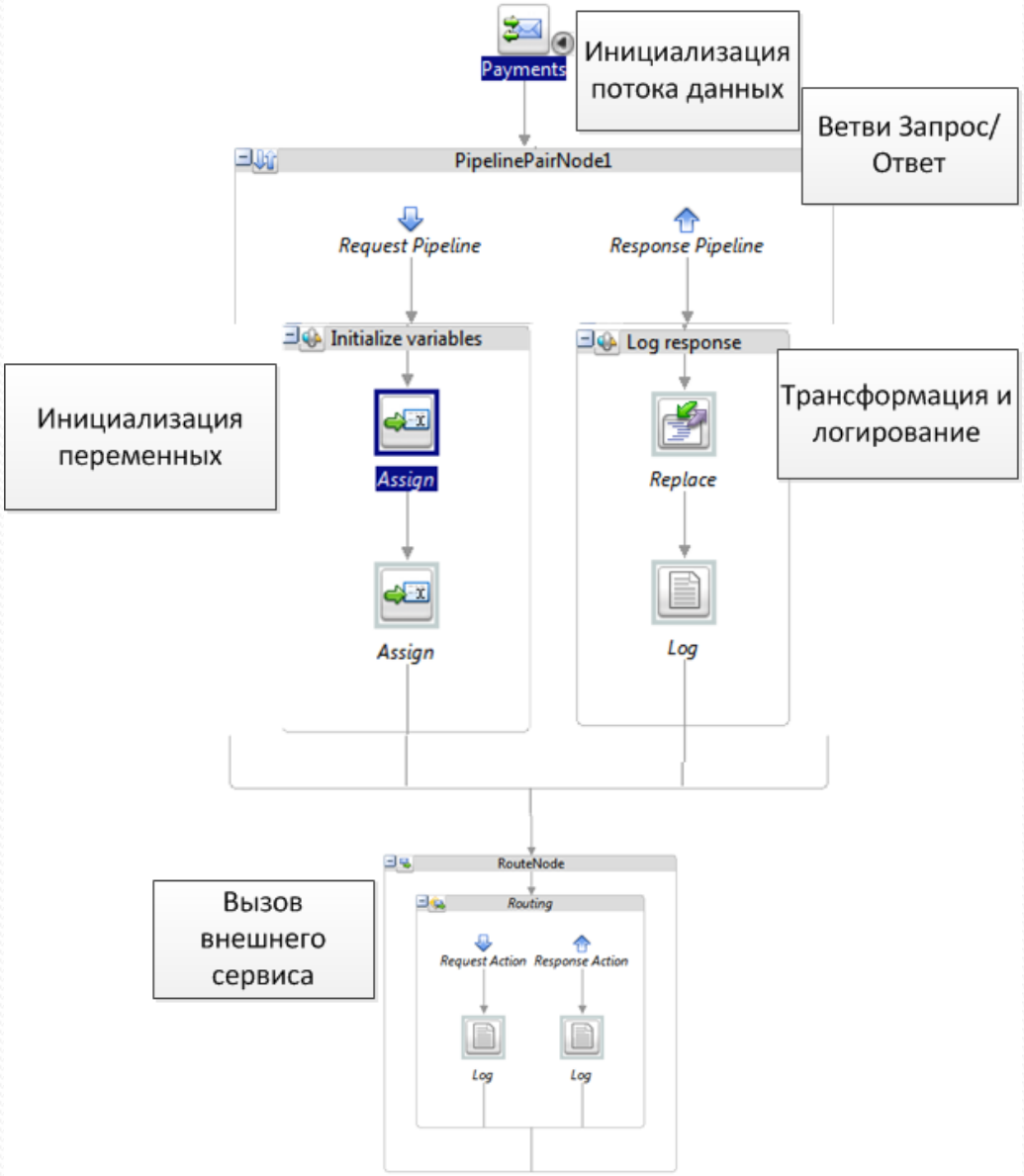
Предназначена для реализации синхронного взаимодействия между потребителем и провайдером сервиса. Данная технология не содержит средств реализации долгоживущих процессов и асинхронных сервисов.

Отвечает за:

- передачу сообщений между бизнес-доменами (это могут быть как подразделения предприятия, так и несколько подразделений, объединенных по функциональному признаку)
- передачу сообщений между композитными приложениями.



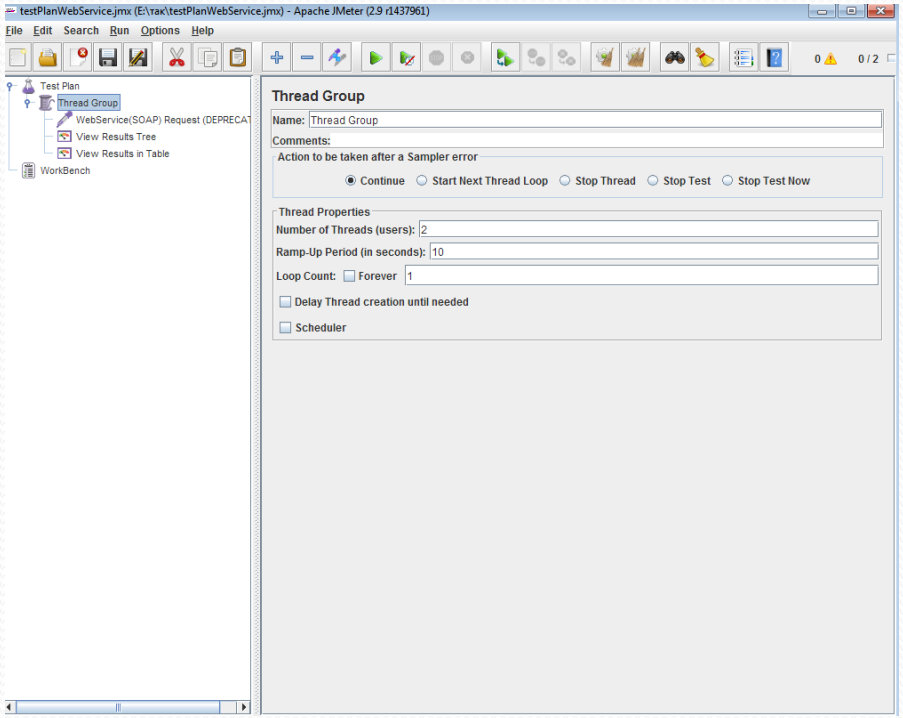
Маршрутизация с использованием OSB:

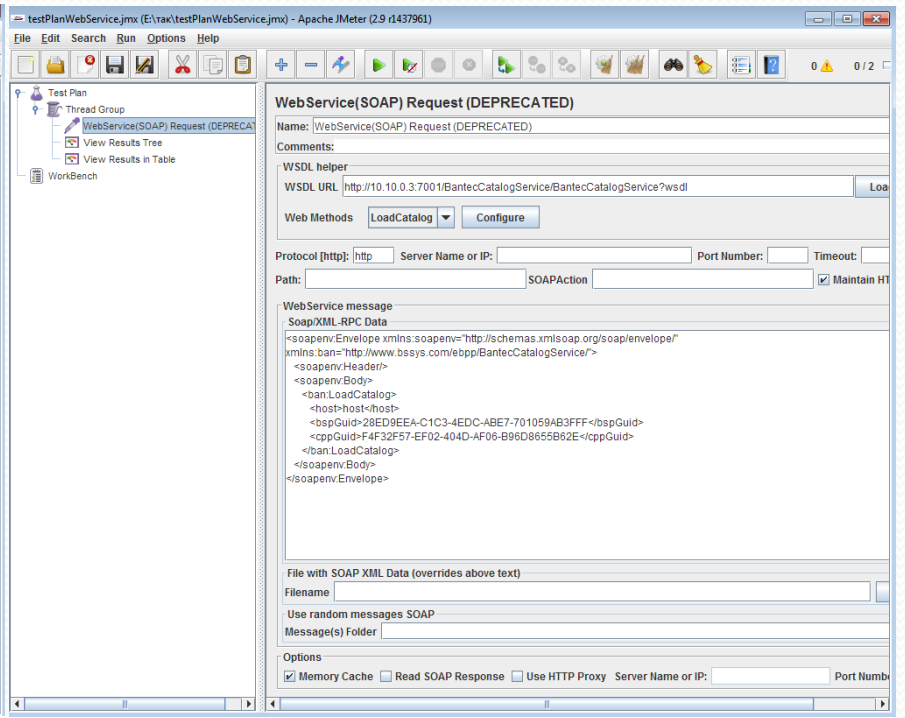


Тестирование системы:

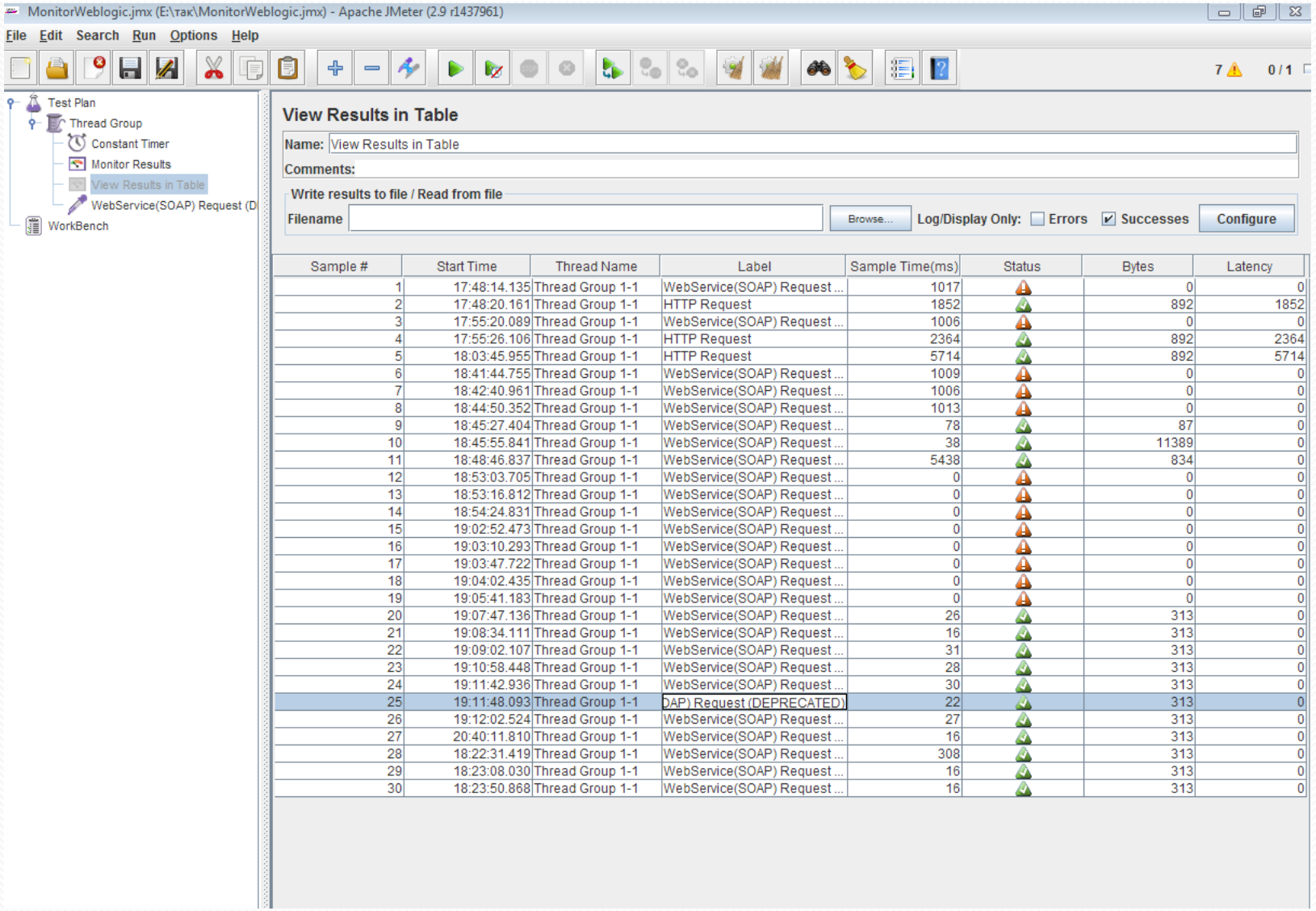
1. Тест-план (функциональное + нагрузочное тестирование)
2. Тест-проект в JMeter:

- Тестовый запрос на сервис загрузки каталога
- Тестовый запос на проверку реквизитов платежа
- Тестовый запрос на проведение платежа (с тем же guid платежа что и при проверке)





Тестирование (результаты):



Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Latency
1	17:48:14.135	Thread Group 1-1	WebService(SOAP) Request ...	1017		0	0
2	17:48:20.161	Thread Group 1-1	HTTP Request	1852		892	1852
3	17:55:20.089	Thread Group 1-1	WebService(SOAP) Request ...	1006		0	0
4	17:55:26.106	Thread Group 1-1	HTTP Request	2364		892	2364
5	18:03:45.955	Thread Group 1-1	HTTP Request	5714		892	5714
6	18:41:44.755	Thread Group 1-1	WebService(SOAP) Request ...	1009		0	0
7	18:42:40.961	Thread Group 1-1	WebService(SOAP) Request ...	1006		0	0
8	18:44:50.352	Thread Group 1-1	WebService(SOAP) Request ...	1013		0	0
9	18:45:27.404	Thread Group 1-1	WebService(SOAP) Request ...	78		87	0
10	18:45:55.841	Thread Group 1-1	WebService(SOAP) Request ...	38		11389	0
11	18:48:46.837	Thread Group 1-1	WebService(SOAP) Request ...	5438		834	0
12	18:53:03.705	Thread Group 1-1	WebService(SOAP) Request ...	0		0	0
13	18:53:16.812	Thread Group 1-1	WebService(SOAP) Request ...	0		0	0
14	18:54:24.831	Thread Group 1-1	WebService(SOAP) Request ...	0		0	0
15	19:02:52.473	Thread Group 1-1	WebService(SOAP) Request ...	0		0	0
16	19:03:10.293	Thread Group 1-1	WebService(SOAP) Request ...	0		0	0
17	19:03:47.722	Thread Group 1-1	WebService(SOAP) Request ...	0		0	0
18	19:04:02.435	Thread Group 1-1	WebService(SOAP) Request ...	0		0	0
19	19:05:41.183	Thread Group 1-1	WebService(SOAP) Request ...	0		0	0
20	19:07:47.136	Thread Group 1-1	WebService(SOAP) Request ...	26		313	0
21	19:08:34.111	Thread Group 1-1	WebService(SOAP) Request ...	16		313	0
22	19:09:02.107	Thread Group 1-1	WebService(SOAP) Request ...	31		313	0
23	19:10:58.448	Thread Group 1-1	WebService(SOAP) Request ...	28		313	0
24	19:11:42.936	Thread Group 1-1	WebService(SOAP) Request ...	30		313	0
25	19:11:48.093	Thread Group 1-1	DAP) Request (DEPRECATED)	22		313	0
26	19:12:02.524	Thread Group 1-1	WebService(SOAP) Request ...	27		313	0
27	20:40:11.810	Thread Group 1-1	WebService(SOAP) Request ...	16		313	0
28	18:22:31.419	Thread Group 1-1	WebService(SOAP) Request ...	308		313	0
29	18:23:08.030	Thread Group 1-1	WebService(SOAP) Request ...	16		313	0
30	18:23:50.868	Thread Group 1-1	WebService(SOAP) Request ...	16		313	0

Выводы

- Проведено предпроектное исследование и сравнение различных архитектур.
- На базе интеграционной системы спроектирован модуль взаимодействия с платежной системой «Рапида»
- С помощью диаграмм UML на этапе проектирования показаны основные компоненты системы
- Разработаны диаграммы классов, последовательности, состояний проектируемой подсистемы.
- Написана реализация модуля на языке java + OSB в соответствии со спецификацией JavaEE
- Произведены отладка и тестирование подсистемы, исправление ошибок
- Подсистема введена в эксплуатацию и тестируется

Спасибо за внимание!