

« Параллельные вычисления »

Н.Е.Богданов

15 мая 2016 г.

Содержание

1	Задание	2
2	Решение	3
2.1	Постановка задачи	3
2.2	Роли	3
2.3	Подробное описание вариантов использования	4
2.4	Use - case диаграмма	5
2.5	Статическая модель предметной области (uml диаграмма классов)	6
2.6	Диаграмма последовательностей	7
2.7	Слой бизнес-логики	8
2.8	Слой источников данных	10
2.9	Сервисный слой и слой представления	14
3	Заключение	16

1 Задание

1. Постановка задачи
 - Описание назначения проектируемой системы
 - Функциональные требования (текстовое описание Участников и их Интересов)
 - Описание бизнес-процессов (этапы, Участники)
2. Разработка вариантов использования (обобщенная диаграмма(ы) прецедентов для всех ролей)
3. Подробное описание всех вариантов использования (текстовое описание с альтернативами)
4. Разработка статической объектной модели предметной области (диаграмма классов)
5. Разработка динамической объектной модели предметной области (диаграмма последовательности)
6. Проектирование слоя бизнес-логики (выбор архитектурного шаблона уровня бизнес-логики)
7. Реализация слоя бизнес-логики (Java, NetBeans), unit-тестирование (JUnit), вместо слоя хранения - шаблон "Репозиторий"
8. Проектирование слоя источников данных (выбор архитектурного шаблона уровня доступа к данным: DB + внешний сервис)
9. Реализация слоя источников данных (JavaDB, NetBeans), unit-тестирование
10. Проектирование сервисного слоя и слоя представления: GUI (Swing), внешний сервис
11. Реализация слоев представления, сервисного слоя, unit-тестирование сервисного слоя
12. Комплексное тестирование системы
13. Пояснительная записка (включает все разделы, указанные выше, а также выводы)

2 Решение

2.1 Постановка задачи

Заказ услуг по строительным работам. Создание и управления заказами на строительные работы, а так же учёт требуемых ресурсов на складе.

2.2 Роли

- Клиент
 1. Заказывает работу.
 2. Принимает результат.
 3. Оплачивает работу.
- Менеджер
 1. Составляет смету + смету доработок (на основе списка доработок от Прораба).
 2. Ведёт учёт ресурсов со склада.(дозаказывает по мере необходимости).
 3. Ведёт учёт бюджета компании.
 4. Принимает оплату от клиента.
- Прораб
 1. Получает список работ.
 2. Выполняет работу.
 3. Составляет список доработок.
 4. Отдаёт работу на приём Клиенту.

2.3 Подробное описание вариантов использования

1. Процесс оформления заказа. прописываются все требуемые ресурсы и услуги оказываемые прорабом. Если ресурсов на складе не хватает то происходит дополнительный заказ ресурсов.

2. Процесс сдачи/приёма работы.

2 Варианта:

- Первый. Успешная сдача объекта клиент принимает работу прораба и получает смету (составленную менеджером) со списком проведённых работ.
- Второй. В случае если клиент требует доработки, прораб составляет список требуемых работ и(или) ресурсов), а менеджер составляет смету доработок, после чего клиент оплачивает 85% от текущего заказа без сметы доработок. а дальше выполняются действия как в первом варианте или повторные доработки.

3. Процесс оплаты счёта. 2 варианта действий, клиент может оплатить:

- наличными или
- по безналичному расчёту.

2.4 Use - case диаграмма

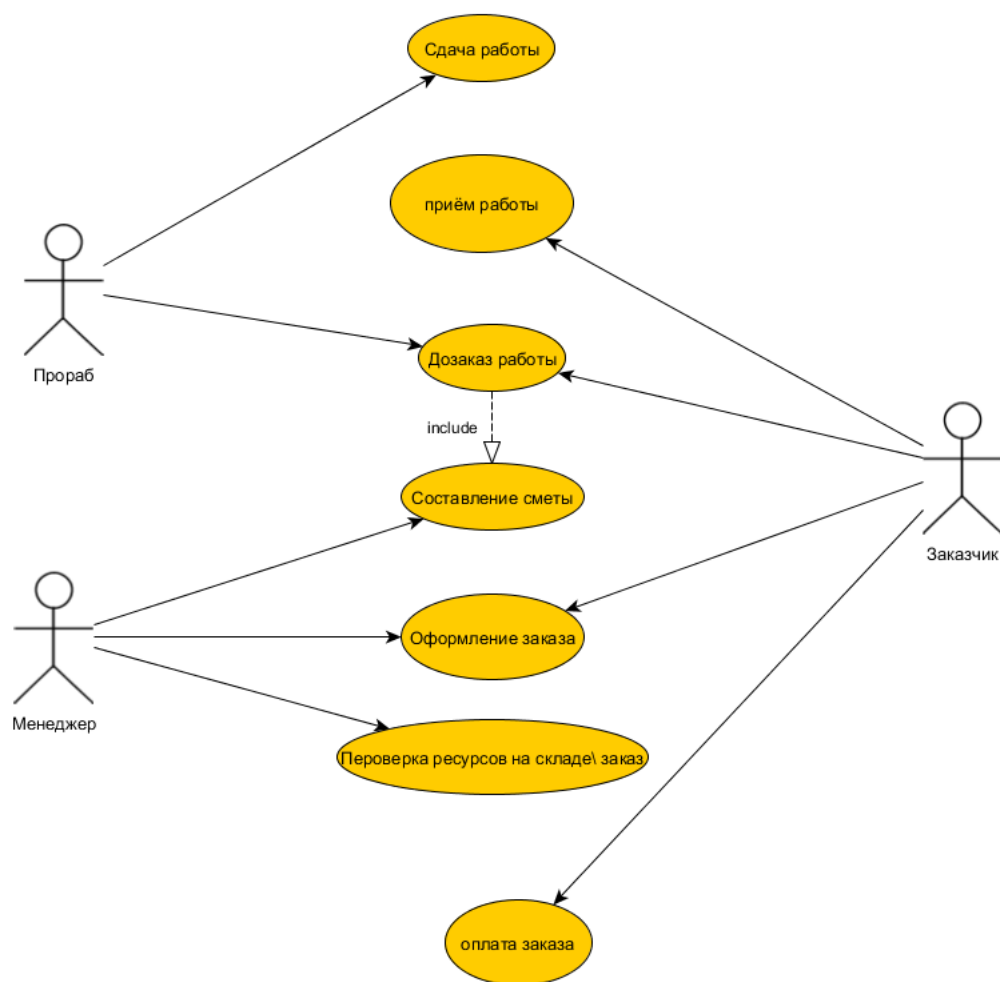


Рис. 1:

2.5 Статическая модель предметной области (uml диаграмма классов)

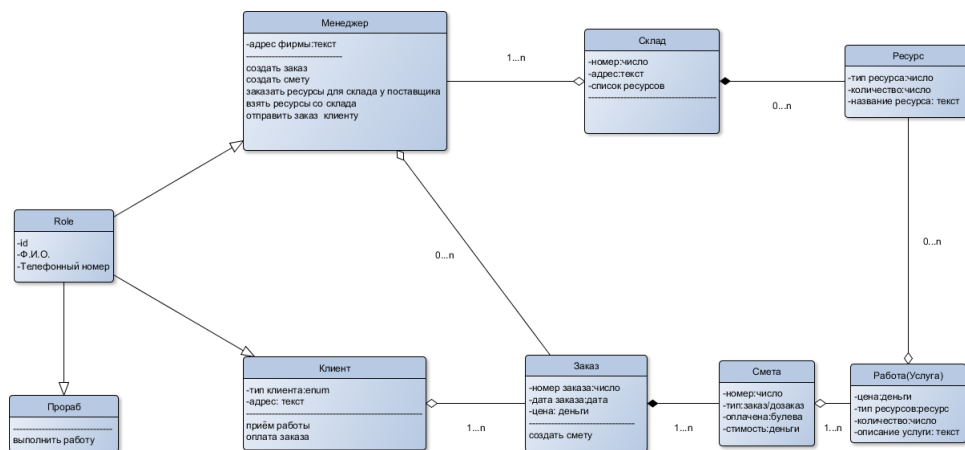
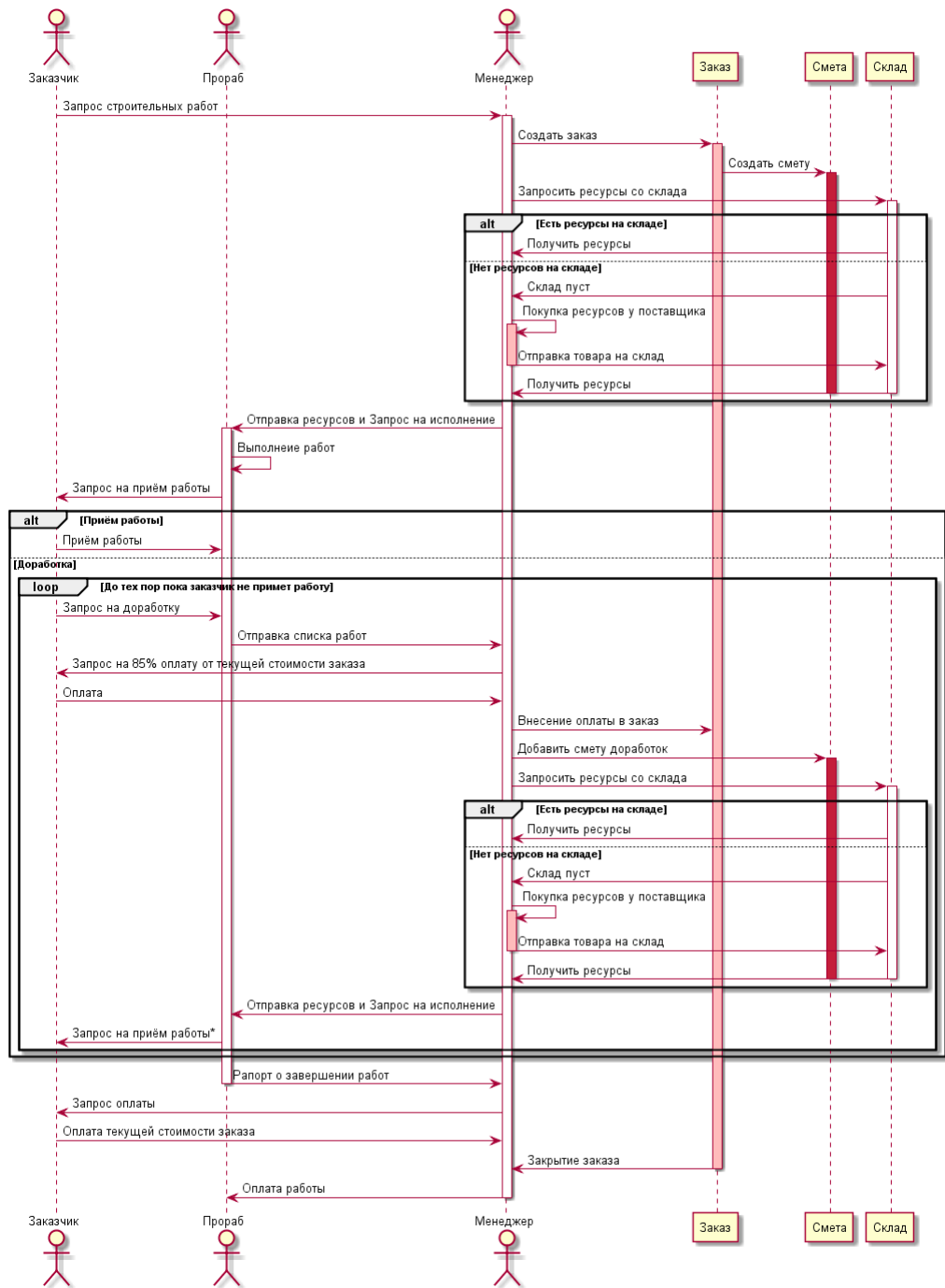


Рис. 2:

2.6 Диаграмма последовательностей



codeuml.com

Рис. 3:

2.7 Слой бизнес-логики

В качестве типового решения бизнес-логики была выбрана: Модель предметной области(Domain Model). Классы бизнес логики соответствуют uml диаграмме (рис.2).

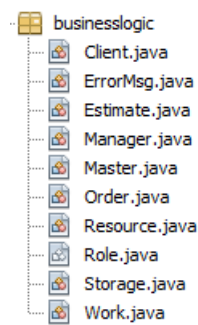


Рис. 4:

Сущности объекты с которыми работают роли:

- Order - Заказ
- Estimate - Смета в заказе
- Storage - Склад содержащий список с ресурсами.
- Work - Работы содержащие список ресурсов
- Resource - Ресурсы.

Роли:

- Role - абстрактное представление роли.
- Manager - Роль менеджера.
- Client - Роль заказчика
- Master - Роль прораба

Единственный элемент являющийся вспомогательным ErrorMsg - Содержит код ошибки и сообщение об ошибке создан специально для обработки ошибок от склада для роли менеджера.

unit-тестирование (JUnit) бизнес логики:

businesslogic















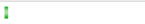

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed Cxty	Missed Lines	Missed Methods	Missed Classes
Manager		100%		100%	0 35	0 77	0 9	0 1
Order		100%		100%	0 50	0 94	0 33	0 1
Work		100%		100%	0 33	0 56	0 19	0 1
Estimate		100%		100%	0 32	0 54	0 22	0 1
Storage		100%		100%	0 25	0 43	0 13	0 1
Resource		100%		100%	0 18	0 39	0 11	0 1
Client		100%	n/a	n/a	0 8	0 14	0 8	0 1
Role		100%	n/a	n/a	0 7	0 14	0 7	0 1
Master		100%	n/a	n/a	0 2	0 4	0 2	0 1
ErrorMsg		100%	n/a	n/a	0 1	0 4	0 1	0 1
Total	0 of 1 333	100%	0 of 167	100%	0 211	0 399	0 125	0 10

Рис. 5:

2.8 Слой источников данных

В качестве решения типового решения источников данных выбран: Преобразователь данных (**Data Mapper**), также дополнительно для облегчения работы были упрощенна работа с SQL - выражениями по средством добавления паттерна строитель для построения SQL - запросов.(класс QueryBilder),а так - же отделения работы с базой данных в класс DatabaseManager.

Для каждого объекта создан свой Mapper с соответствующей приставкой в названии (пример OrderMapper для объекта Order из слоя бизнес логики) некоторые используют внутри Mapper'ы если объекты составные.

В качестве хранилища данных выбрана СУБД Firebird.

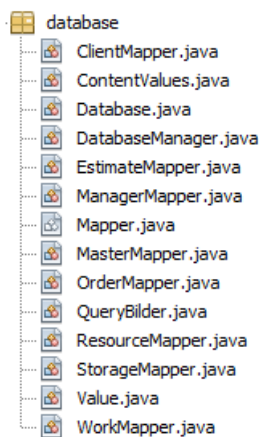


Рис. 6:

Представление таблиц данных в БД.

Таблицы:

- StorageInformation - информация о складе.
- Storage - информация о ресурсах на складе и их количество.
- Resource - информация о ресурсе.
- Work - информация о работе.
- WorksAndResource - информация о ресурсах и их количестве нужном для работы.
- Estimate - информация о списке смет для заказов.
- EstimateWorks - информация о списке работ для смет.
- Manager - информация о менеджерах.
- Master - информация о прорабах.
- Client - информация о заказчиках.

Дополнительно созданы View для упрощения запросов к данным сущностей бизнес логики распределённым между несколькими таблицами (склады, работы, сметы):

- StorageView
- WorkView
- EstimateView

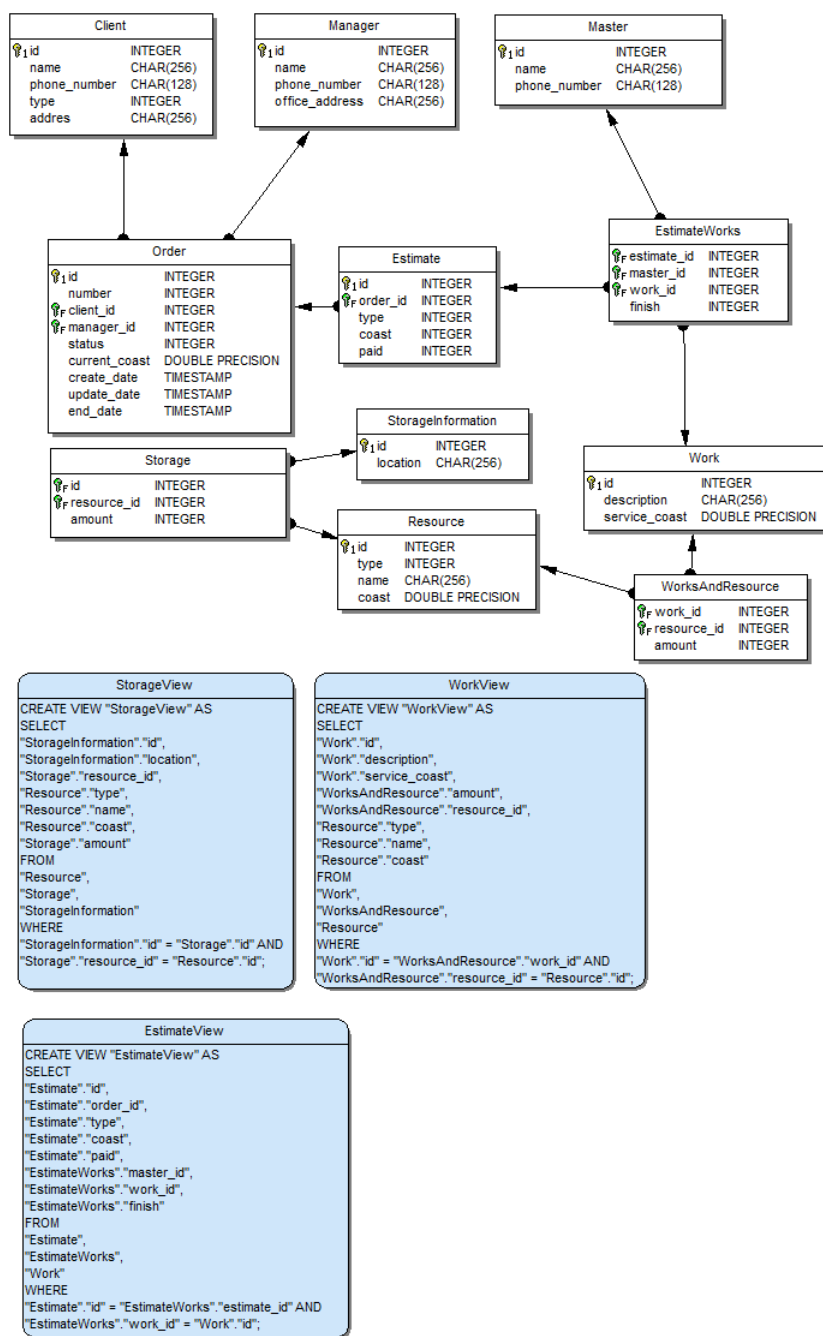


Рис. 7:

unit-тестирование (JUnit) слоя источников данных:

database

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
DatabaseManager		44%		57%	15	28	70	121	11	21	0	1
EstimateMapper		86%		71%	10	30	17	123	1	9	0	1
StorageMapper		89%		71%	8	23	15	110	2	9	0	1
WorkMapper		92%		79%	6	23	10	113	1	9	0	1
DatabaseManager.DriverType		0%		n/a	4	4	3	3	4	4	1	1
ClientMapper		92%		65%	8	19	6	69	1	9	0	1
ManagerMapper		92%		65%	8	19	6	67	1	9	0	1
MasterMapper		92%		65%	8	19	6	66	1	9	0	1
ResourceMapper		93%		67%	7	18	4	65	1	9	0	1
OrderMapper		98%		92%	3	30	3	143	0	11	0	1
DatabaseManager.CharEncoding		98%		n/a	2	4	0	41	2	4	0	1
DatabaseManager.IsolationLevel		81%		n/a	2	4	0	4	2	4	0	1
Database.Master		0%		n/a	1	1	1	1	1	1	1	1
Database.Client		0%		n/a	1	1	1	1	1	1	1	1
Database.Work		0%		n/a	1	1	1	1	1	1	1	1
Database.WorksAndResource		0%		n/a	1	1	1	1	1	1	1	1
Database.Resource		0%		n/a	1	1	1	1	1	1	1	1
Database.StorageView		0%		n/a	1	1	1	1	1	1	1	1
Database.Roles		0%		n/a	1	1	1	1	1	1	1	1
Database.Storage		0%		n/a	1	1	1	1	1	1	1	1
Database.Manager		0%		n/a	1	1	1	1	1	1	1	1
Database.EstimateView		0%		n/a	1	1	1	1	1	1	1	1
Database.EstimateWorks		0%		n/a	1	1	1	1	1	1	1	1
Database.StorageInformation		0%		n/a	1	1	1	1	1	1	1	1
Database.Estimate		0%		n/a	1	1	1	1	1	1	1	1
Database.WorkView		0%		n/a	1	1	1	1	1	1	1	1
Database.Order		0%		n/a	1	1	1	1	1	1	1	1
Database		0%		n/a	1	1	2	2	1	1	1	1
QueryBuilder		100%		100%	0	26	0	67	0	6	0	1
ContentValues		100%		100%	0	11	0	13	0	10	0	1
Value		100%		n/a	0	1	0	4	0	1	0	1
Mapper		100%		n/a	0	1	0	1	0	1	0	1
Total	695 of 5 778	88%	62 of 270	77%	97	276	156	1 026	43	141	17	32

Рис. 8:

Не протестированными остались классы содержащие только статические имена полей таблиц и названий таблиц в базе данных, а также метод saveAll для нескольких классов, но был позже протестирован при сборке проекта с GUI.

2.9 Сервисный слой и слой представления

Графика была создана при помощи визуального редактора в NetBeans. Конфигурационный файл, Swing GUI

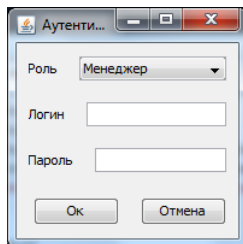


Рис. 9: Окно авторизации

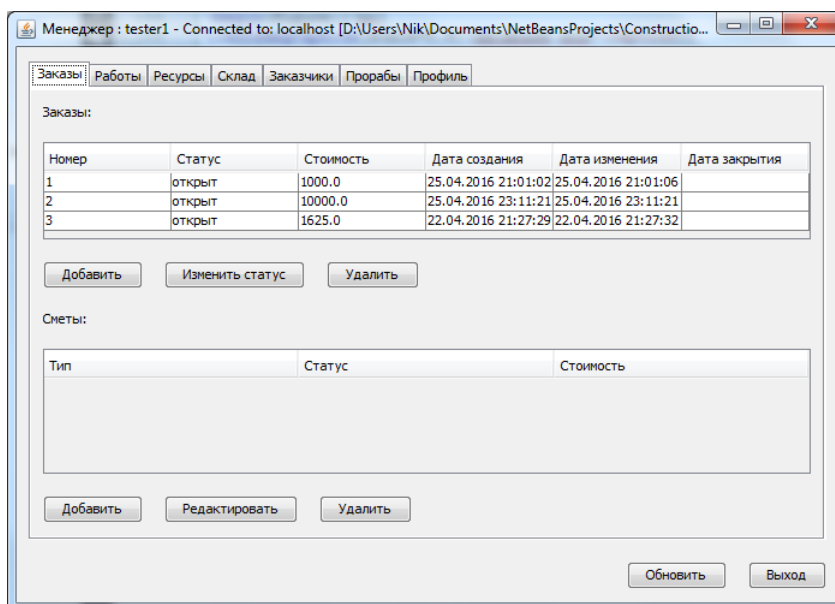


Рис. 10: Графическое представление для роли Менеджера

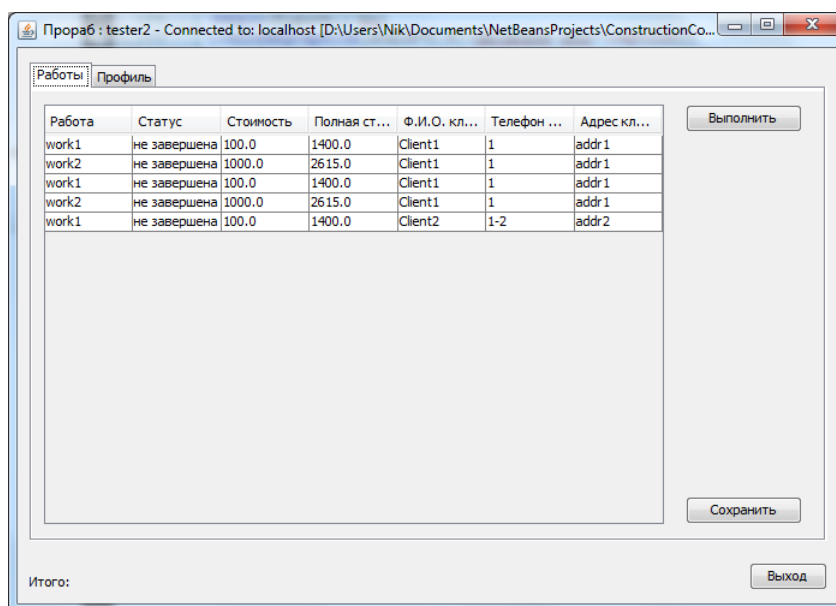


Рис. 11: Графическое представление для роли Прораба

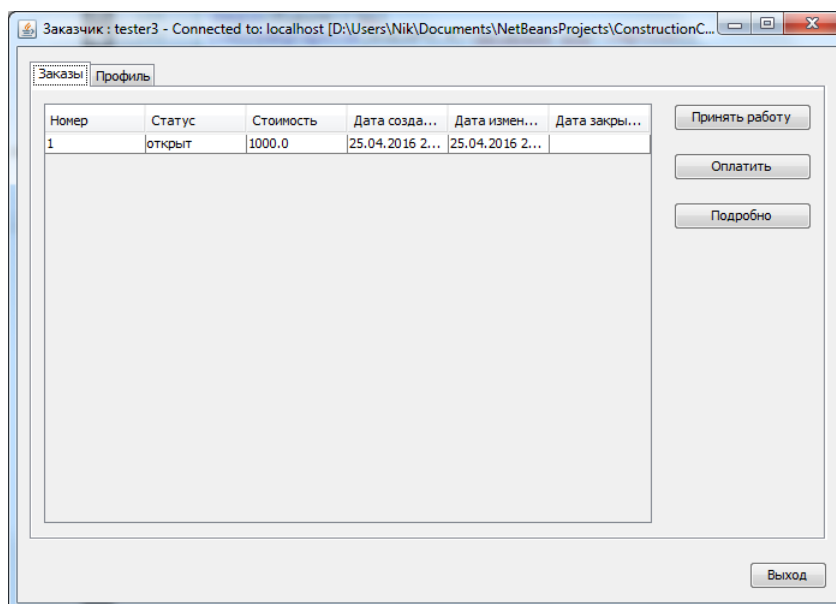


Рис. 12: Графическое представление для роли Заказчика

Тестирование графики проводилось полностью при сборке работы всего проекта.

Тестирование конфигурационного файла:

service.config

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed Cxty	Missed Lines	Missed Methods	Missed Classes
Config		71%		50%	5 12	2 20	0 5	0 1
ConfigXmlParser		72%	n/a		0 3	3 12	0 3	0 1
ConfigHandler		98%		97%	3 42	2 96	1 7	0 1
ConfigDatabase		98%		50%	12 27	1 36	0 15	0 1
ConfigItem		97%		50%	5 14	1 24	0 9	0 1
ConfigRole		95%		50%	6 14	1 19	0 8	0 1
ConfigSettings		100%		100%	0 8	0 14	0 6	0 1
Total	67 of 966	93%	32 of 134	76%	31 120	10 221	1 53	0 7

Рис. 13:

3 Заключение

Данная работа моделирует пример проектирование архитектуры информационной системы для бизнес процессов на примере строительной организации.