МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра вычислительной техники

УТВЕРЖДАЮ
Проректор по учебной работе
_____ О.Г. Локтионова
«25» 02 2022 г.

IT-стандарты

Методические указания к лабораторным работам
для студентов направления подготовки 09.03.01

Курск 2022

УДК 004

Составители: А.В. Киселев

Рецензент
Кандидат технических наук, доцент *Ю.А. Халин*

IT-стандарты: методические указания к лабораторным работам для студентов направлений подготовки 09.03.01/ Юго-Зап. гос. ун-т; сост.; А.В. Киселев. – Курск, 2022. - 59 с.: - ил. 14 , табл. 4.– Библиогр.: с. 59

Содержат сведения о стандартах в области информационных систем, комплексе нормативных документов на автоматизированные системы, комплексе стандартов Единой системы программной документации, стандартах в области информационной безопасности ИТ.

Предназначены для студентов направления подготовки 09.03.01 очной формы обучения.

Методические указания соответствуют рабочей программе дисциплины «IT-стандарты».

Текст печатается в авторской редакции

Подписано в печать . Формат 60*84 1/16.
Усл. печ. л.___. Уч.-изд. л. 3,1. Тираж 50 экз. Заказ 898 . Бесплатно.
Юго-Западный государственный университет.
305040 Курск, ул. 50 лет Октября, 94.

Лабораторная работа №1
Стандарты предпроектного обследования

1.1. Цель работы

Исследование методики предпроектного обследования предприятия. Изучение методов обследования и сбора информации. Приобретение навыков анализа полученных материалов для последующего моделирования.

1.2. Основные теоретические положения

Целью анализа предметной области является рассмотрение существующего состояния предметной области, характеристик объекта и системы управления, выявление и анализ проблем предметной области, наличие компьютеризированных информационных технологий, составсредств компьютерной техники и программного обеспечения, оценка их достаточности и эффективности для решения задач информатизации (автоматизации).

В данном разделе следует отразить следующие характеристики выбранной предметной области:

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ
- 1.1 Организационно-экономическая характеристика компании
- 1.2 Экономический анализ компании, выявление и анализ целей и проблем
- 1.3 Бизнес- и информационные процессы компании
- 1.4 ИТ-инфраструктура компании
- 1.5 Постановка задачи проектирования ИС

В качестве предметной области (объекта автоматизации или информатизации) может выступать:

- организация или ее подразделение (предприятие, учреждение, фирма, и т. п.),
- отдельный вид деятельности (бизнес-процесс).

Организационно-экономическая характеристика предметной области должна включать:

– наименование, организационная форма, юридический статус и миссию организации (необходимо выяснить миссию организации, оценить правильность ее формулировки и, если надо, дать свою формулировку),

– его организационную схему компании (с указанием общей численности работающих), пример организационной схемы приведен на рис.1;

– краткую характеристику технико-экономических аспектов подразделений.

Аспектами описания компании являются: основные задачи; тип производства (услуг); номенклатура готовой продукции (услуг); номенклатура материалов и ресурсов.

Необходимо установить базовые экономические и другие показатели, характеризующие деятельность организации (например, прибыль, рентабельность, число обслуживаемых клиентов, и т. п.).

Пример целей и задач компании приведен на рис.1.

Рисунок 1 – Пример целей и задач компании

Пример организационной схемы компании представлен на рис.2.

При анализе проблем предметной области следует сделать акцент на проблемах и недостатках, устранение которых предполагается осуществить в проекте, например:

- невозможность расчета показателей, необходимых для управления объектом из-за сложности вычислений или большого объема информации;
- высокая трудоемкость обработки информации (привести объемно-временные параметры);
- низкая оперативность, снижающая качество управления объектом;
- невысокая достоверность результатов решения задачи из-за дублирования потоков информации;
- несовершенство процессов сбора, передачи, обработки, хранения, защиты целостности и секретности информации и процессов выдачи результатов расчетов конечному пользователю и т. д.

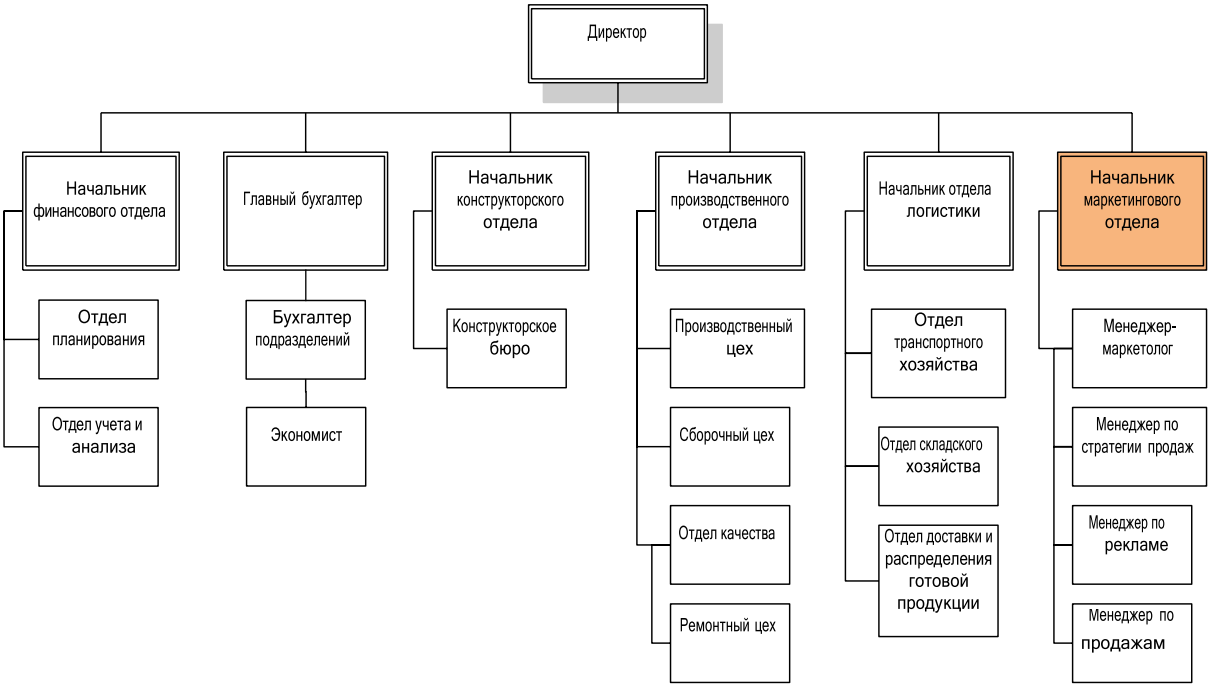


Рисунок 2 – Пример организационной схемы

В области информационного обеспечения управления рассматриваются потребности субъекта управления в экономической и другой информации для принятия управленческих решений, а также имеющиеся проблемы информационного обеспечения. Необходимо описать источники информации, способы ее хранения, передачи и переработки (используя методологию DFD). При этом следует:

- описать существующую технологию выполнения выбранной для рассмотрения функции управления (или комплекса функций), т. е. указать на следующее: особенности расчета показателей; перечни и источники используемых входных документов; методы и технические средства, применяемые для их обработки; методы защиты конфиденциальной информации (сведений, составляющих коммерческую или производственную тайну);
- выявить основные недостатки, присущие существующей практике управления и обработки экономической информации.

Описание задачи должны быть выполнено в виде единого связного текста и может сопровождаться диаграммами структурного системного анализа и обобщающими таблицами или разъясняющими схемами.

Для выполнения анализа объекта управления и решаемой задачи рекомендуется использовать методологии IDEF0, IDEF3, DFD, UML.

При описании ИТ-инфраструктура организации необходимо:

– идентифицировать существующие ИС и описать бизнес-процессы, которые они поддерживают;
– дать описание сетевой архитектуры, компьютерной техники и средств телекоммуникаций;
– описать системное и прикладное программное обеспечение;
– описать работу ИТ-подразделений и служб.

Сетевая архитектура. Сетевая архитектура представляет собой множество технических средств: сервера, клиентские устройства доступа, каналы связи. Необходимо рассмотреть, в случае наличия, существующую локальную вычислительную сеть, оборудование, структурированную кабельную сеть и ее атрибуты. Необходимо указать наличие доступа к внешним телекоммуникациям (в частности, выход в Internet), параметры подключения.

Примеры сетевой архитектуры представлены на рис. 3.

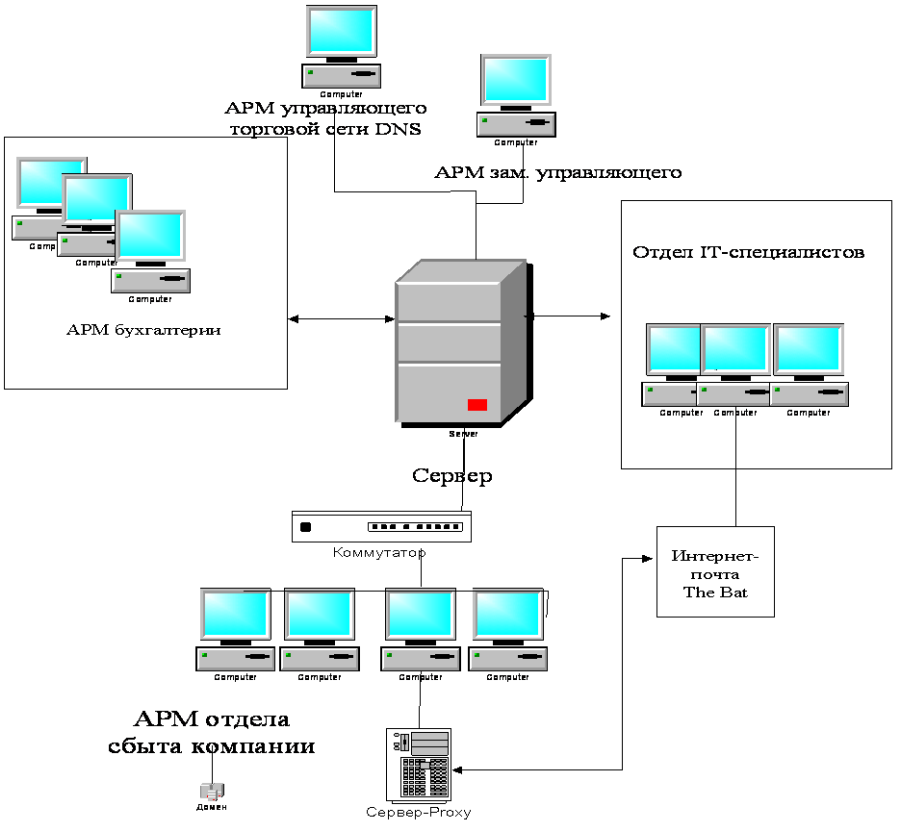


Рисунок 3 - Пример сетевой архитектуры

Компьютерные средства. Необходимо кратко описать компьютерные средства, используемые в организации (подразделении). Примером такого описания может быть следующая таблица:

Компьютерная техника	Количество
Компьютеры всего	**12**
из них: компьютеры в ЛВС	8
серверы	2
несвязанных ЛВС	1
Принтеры	**5**
из них лазерные	4

Далее следует привести обобщенные параметры компьютеров, например:

Наименование	Цена, руб.	Количество, ед.	Итого, тыс.руб.
монитор ЖК 17''LG L1752 H-BF Flatron (LCD 1280x1024)	6306	10	63.060
Сист.блок Futjitsu P4-630-3.0 Ггц/DDR 512Mb-2*256Mb/80Gb	16276	10	162.760
Клавиатура Logitech Deluxe 250 Y-SAF76	276	10	2.760
Мышь Logitech Optical<M-SBF96>	296	10	2.960
Стоимость установки и обучения Итого	23154		23.154 16276

Программная архитектура. Программную архитектуру целесообразно формировать исходя из существующих программных продуктов, которые функционируют на предприятии. (рис. 3).

Необходимо указать имеющиеся специализированные или офисные программы, бухгалтерские, складские и другие информационные системы. Необходимо показать, для решения каких задач они используется, в виде таблицы:

Таблица 1 – Перечень программных продуктов, используемых компанией

№ п/п	Программа	Решаемая задача
1	Microsoft Office 2007	Ведение текущей документации, подготовка отчетов для вышестоящей организации
2	1С:Предприятие 8. 1С-Логистика:Управление перевозками»	Специализированное тиражное решение на платформе «1С:Предприятие 8» для автоматизации процессов транспортировки грузов.
3	…	…

Или в виде схемы (рис.4).

В заключение (или в каждом подразделе) надо провести анализ и сделать выводы о достаточности и эффективности использования имеющихся программных средств и компьютерного оборудования

При выполнении анализа системы обеспечения информационной безопасности и защиты информации следует отметить имеющиеся решения по политике безопасности в компании, а также программные и аппаратные средства информационной безопасности (ИБ) и защиты информации (ЗИ), если эти методы и средства используются, то каким образом.

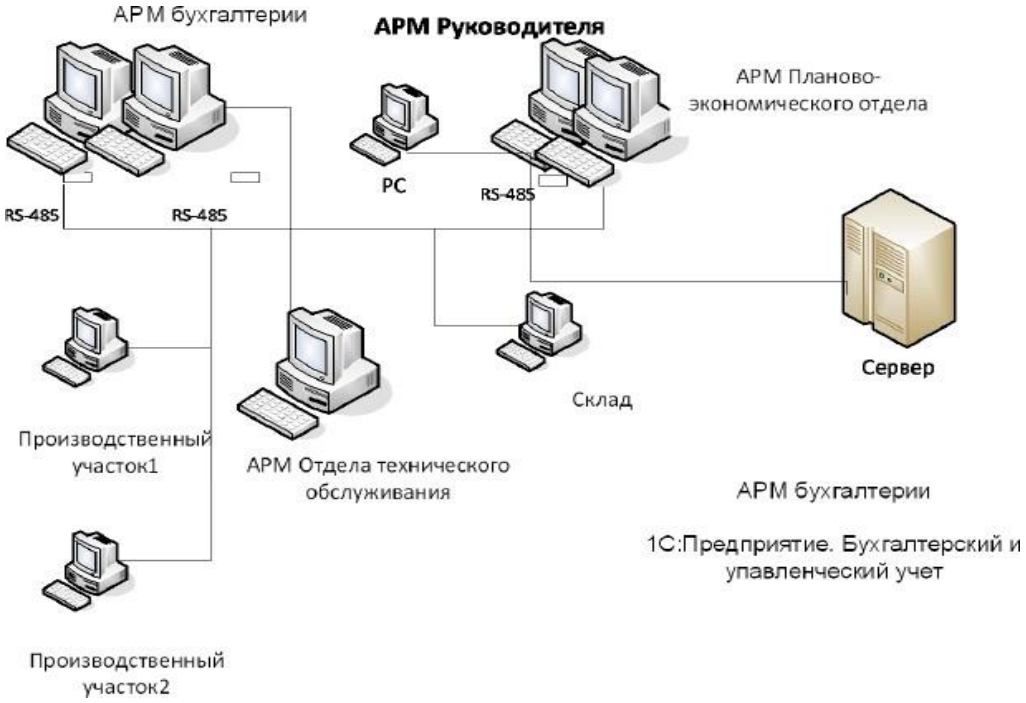


Рисунок 4 – Схема программных продуктов,
используемых на предприятии

При анализе системы и имеющихся в ней методов и средств ИБ и ЗИ необходимо отразить:

а) существующую в компании политику безопасности (нормативно-правовые и организационно-распорядительные документы, регламенты, процедуры, должностные инструкции и т. д.), рекомендуется указать основные положения политики безопасности (регламенты использования сети Internet, электронной почты, доступа к служебной информации, доступа к информации, составляющей коммерческую тайну, установки и использования программного обеспечения);

б) существующие программные и аппаратные средств ИБ и ЗИ, их использование в организации (привести перечень используемых средств, отразив их назначение, параметры и возможности);

в) порядок реализации системы обеспечения ИБ и ЗИ (кто этим занимается, кто отвечает, структура);

г) обеспечение ИБ и ЗИ на различных уровнях: программный, аппаратный, организационный (права доступа, права пользователя системы, парольная защита, доступ к базе, программные средства защиты, встроенные средства защиты, ведение логов и так далее);

д) какие используются средства защиты для Internet-систем от внешних угроз (взлом сайта, нарушение его работы и так далее);

е) какие используются средства защиты от инсайдерских угроз (хищение и порча данных сотрудниками организации, ошибки при пользовании программным и аппаратным обеспечением и так далее).

В заключении необходимо сделать выводы о степени обеспечения ИБ и ЗИ, а также необходимости соответствующих разработок.

Анализ путей решения имеющихся проблем предполагает проведение анализа стратегий, который заключается в выборе целей, которые могут быть достигнуты путем создания информационной системы. При анализе целей и задач проектирования ИС на основании выбранных стратегий необходимо сформулировать цель и задачи разработки проекта. Стоит определить тип проектируемой системы. Это может быть:

- система электронной обработки данных;
- информационно-поисковая система;
- диалоговая система решения задачи или обработки транзакций;
- система поддержки принятия решений;
- автоматизированное рабочее место (АРМ);
- автоматизированная система управления.

Контрольные вопросы

1. В чем заключается методика предпроектного обследования?
2. Какие существуют универсальные методы, пригодные для обследования всех функциональных звеньев предприятия?
3. Какие существуют характеристики документа?
4. Каким образом производится кодирование полученной документации?
5. На каких уровнях проводится обследование аспектов деятельности предприятий?
6. Что включает информационная база данных?
7. В каких направлениях выполняется информационный анализ предметной области?
8. Цель анализа полученной информации.

Приложение 1

Пример описания компании «Furniture of dream»

Мебельная компания «Furniture of dream» ведет свою историю с 1990 г. Компания имеет большой опыт работы в сфере производства и изготовления мебели.

Описание компании

Головной офис компании «Furniture of dream» находится в Москве.

Наша компания занимается производством и изготовлением: шкафов, диванов, кресел, кухонной мебели, тумб, комодов, стульев, офисной мебели, кроватей, прихожих, детской мебели, столов, спальной мебели, гостиной мебели и других видов бытовой мебели.

Компания «Furniture of dream» стремится к созданию стильной мебели на любой вкус и кошелек.

Сегодня «Furniture of dream» - один из ведущих российских производителей и дистрибьюторов бытовой мебели.

В компании работает 3300 сотрудников. Объем продаж составляет 6 052,4 млн. руб. Объем производства мебели в РФ (по данным Ассоциации предприятий мебельной и деревообрабатывающей промышленности России) - 178 252,3 млн. Основные рынки сбыта – Россия, Украина и Белоруссия.

Стратегические цели и задачи

Основная стратегическая цель компании «Furniture of dream» заключается в создании и организации производства новой линейки мебели, которая позволит найти новую категорию потребителей и существенно увеличит доходность продаж.

Миссия компании – Производство качественной мебели по доступным ценам.

К целям компании можно отнести укрепление позиций на рынке и повышение конкурентоспособности.

Перед компанией стоят следующие стратегические цели:

– Укрепление позиций на рынках стран СНГ;

– Обеспечение роста продаж;

– Изыскание новых сегментов сбыта продукции – поиск новых категорий потребителей.

– Для достижения этих стратегических целей необходимо:

– Привлечение новых оптовых покупателей на взаимовыгодных условиях

– Обновление производственных мощностей

– Использование новых видов сырья

– Привлечение на работу новых дизайнеров

– Расширение ассортимента

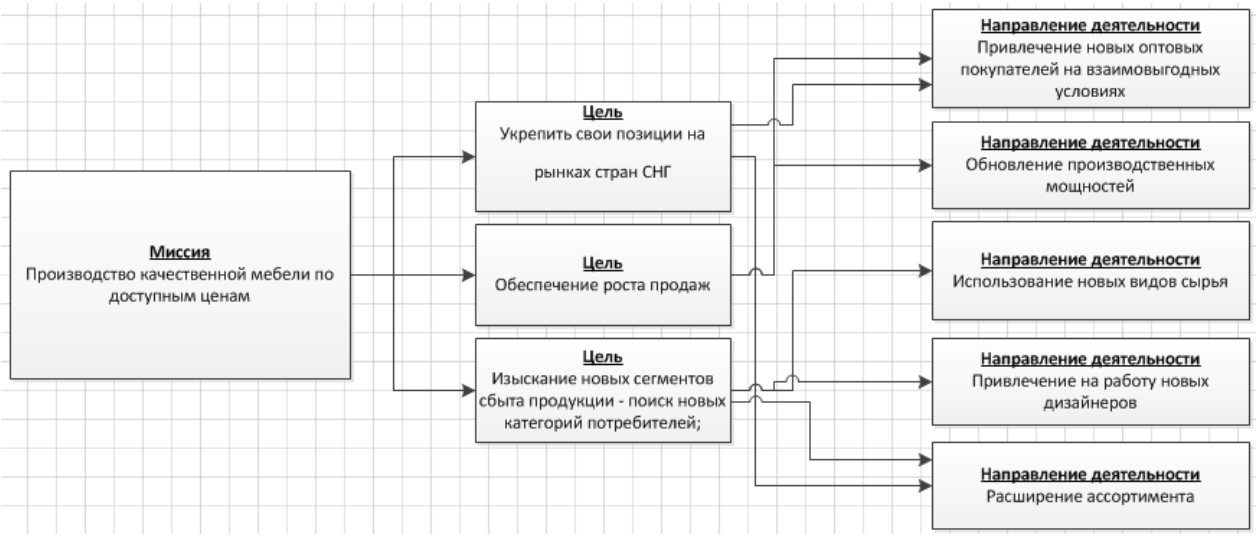


Рис. 1. Стратегические цели и задачи компании «Furniture of dream»

Описание продуктов и услуг

Производимую продукцию модно разделить на 2 вида:

1. Производство мягкой мебели
2. Производство корпусной мебели

Производство мягкой мебели

Мягкая мебель состоит из нескольких основных частей:

– Каркас
– Наполнитель
– Обивка
– Механизм трансформации

Материалы, используемые при производстве мягкой мебели:

Фанера, деревянный брус, ДВП, ДСП, металлический каркас, поролон, синтепон, полиуретан, натуральная и искусственная кожа, шерсть, синтетические мебельные ткани, мебельный велюр.

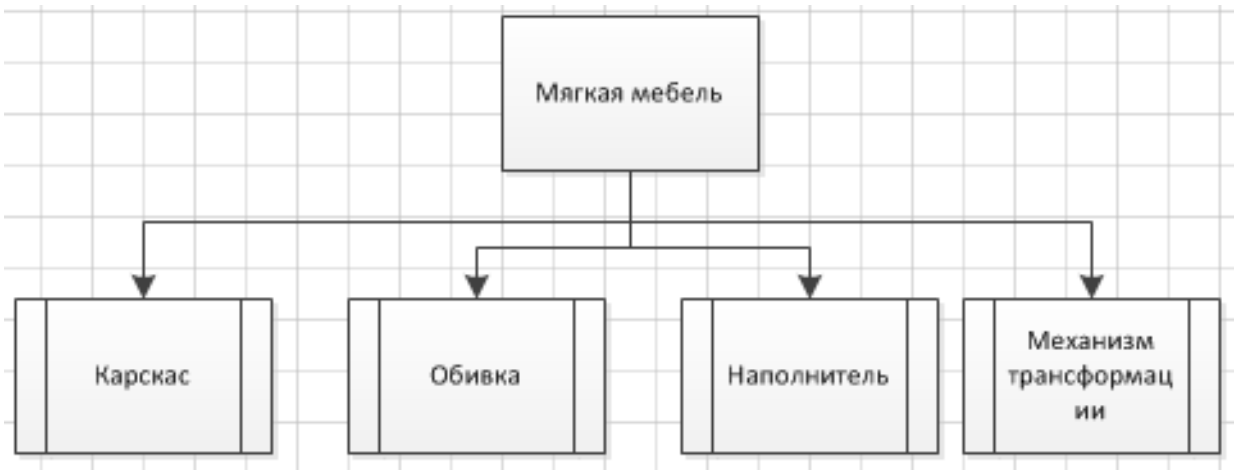


Рис. 2. Модель мягкой мебели.

Производство корпусной мебели

Корпусная мебель состоит из нескольких основных частей:

– Короб
– Фасады
– Цоколь

Материалы, используемые при производстве корпусной мебели:

Деревянный монолит, деревянный щит, шпон, ДСП, ДВП, МДФ, металлический профиль, стекло, зеркало, отделочный пластик, различные виды композитов.

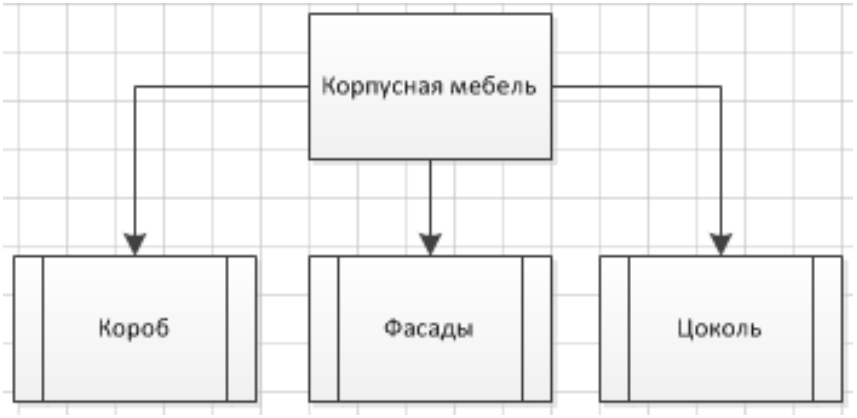


Рис. 3. Модель мягкой мебели.

Организационная структура компании

На рисунке 4 представлена организационная структура компании по производству и изготовлению мебели «Furniture of dream».

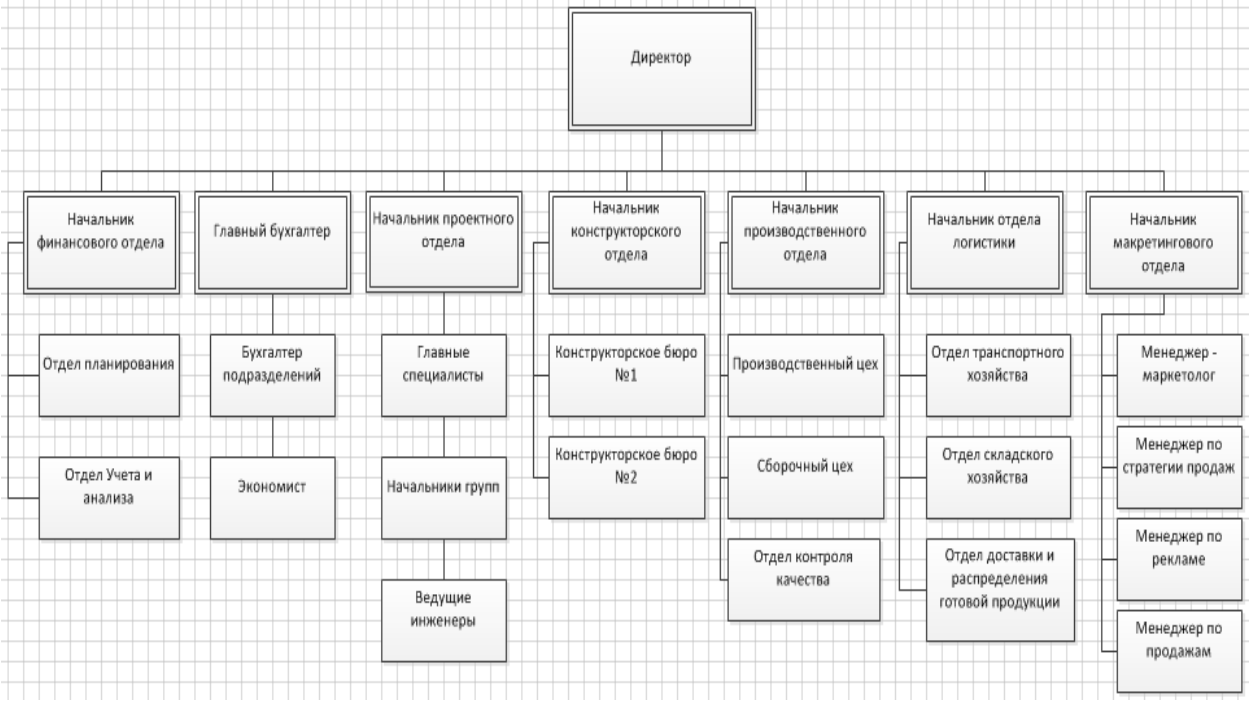


Рис. 4. Модель организационной структуры

Анализ информационного обеспечения компании

В своей деятельности каждый отдел использует специализированное ПО, которое отвечает специфическим требованиям каждого отдела. Бухгалтерия использует программу 1С: Бухгалтерия, проектный отдел - «AutoCAD 2014», конструкторский отдел – «Solid Works 2014», «AutoCAD 2014», производственный отдел – «1С: Предприятие», отдел закупок – «Реестр закупок». Для маркетингового отдела создадим информационную систему.

Основные бизнес-процессы компании

Функции информационной системы маркетингового отдела:

1. Хранить информацию об ассортименте;
2. Помощь в подборе мебели для клиента;
3. Оформление договоров;
4. Формирование отчетности;
5. Хранение сведений о клиентах;
6. Оформление заявок клиента (для заказа товара, отсутствующего на складе).

Лабораторная работа №2
РАЗРАБОТКА ТЕХНИЧЕСКОГО ЗАДАНИЯ

1. Цель работы

Целью лабораторной работы является изучение рекомендаций стандартов на составление технического задания:

- ГОСТ 19.201-78 ЕСПД. Техническое задание. Требование к содержанию и оформлению; ГОСТ 24.201-79 Документация на АСУ. Требование к содержанию документа «Техническое задание»;
- ГОСТ 34.602-89. Информационная технология. Автоматизированные системы. Техническое задание на создание автоматизированной системы;
- IEEE 830-1998 «Методика составления спецификаций требований к программному обеспечению». Получение практических навыков в разработке и структуризации требований в пределах одной проблемной области.

2. Стандарты разработки ГОСТ 19.201-78 и ГОСТ 34.602-89

Техническое задание является исходным материалом для создания информационной системы или другого продукта. Поэтому техническое задание (сокращенно ТЗ) в первую очередь должно содержать основные технические требования к продукту и отвечать на вопрос, что данная система должна делать, как работать и при каких условиях.

Как правило, этапу составления технического задания предшествует проведение обследования предметной области, которое завершается созданием аналитического отчета. Именно аналитический отчет (или аналитическая записка) ложится в основу документа Техническое задание.

Если в отчете требования заказчика могут быть изложены в общем видеи проиллюстрированы UML-диаграммами, в техническом задании следует подробно описать все функциональные и пользовательские требования к системе. Чем подробнее будет составлено техническое задание, тем меньше спорных ситуаций возникнет между заказчиком и разработчиком во время приемочных испытаний.

Таким образом, техническое задание является документом, который позволяет как разработчику, так и заказчику представить конечный продукт и впоследствии выполнить проверку на соответствие предъявленным требованиям.

Руководствующими стандартами при написании технического задания являются ГОСТ 34.602.89 «Техническое задание на создание автоматизированной системы» и ГОСТ 19.201-78 «Техническое задание.

Требования к содержанию и оформлению». Первый стандарт предназначен для разработчиков автоматизированных систем, второй для программных средств (разницу между данными сериями мы обсуждали в статье «Что такое ГОСТ»).

Итак, ниже мы представляем список и описание разделов, которые должно содержать техническое задание согласно ГОСТам.

ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению	**ГОСТ 34.602.89 Техническое задание на создание автоматизированной системы**
1. Введение	1. Общие сведения
2. Основания для разработки	
3. Назначение разработки	2. Назначение и цели создания системы
	3. Характеристика объекта автоматизации
4. Требования к программе или программному изделию	4. Требования к системе
4.1. Требования к функциональным характеристикам	4.2. Требования к функциям (задачам), выполняемым системой
	4.1. Требования к системе в целом
	4.1.1. Требования к структуре и функционированию системы
	4.1.3. Показатели назначения
4.2. Требования к надежности	4.1.4. Требования к надежности
	4. 1.5. Требования к безопасности
	4. 1.6. Требования к эргономике и технической эстетике
4.3. Условия эксплуатации	4.1.2. Требования к численности и квалификации персонала системы и режиму его работы
	4. 1.9. Требования к защите информации от несанкционированного доступа
	4. 1.10. Требования по сохранности информации при авариях
	4. 1.11. Требования к защите от влияния внешних воздействий
	4. 1.12. Требования к патентной чистоте
	4. 1.13. Требования по стандартизации и унификации
4.4. Требования к составу и параметрам технических средств	4. 1.8. Требования к эксплуатации, техническому обслуживанию, ремонту и хранению компонентов системы
4.5. Требования к информационной и программной совместимости	
4.6. Требования к маркировке и упаковке	
4.7. Требования к транспортированию и хранению	4. 1.7. Требования к транспортабельности для подвижных систем
4.8. Специальные требования	4. 1.14. Дополнительные требования
	4.3. Требования к видам обеспечения
5. Требования к программной	8. Требования к документированию

документации	
6. Технико-экономические показатели	
7. Стадии и этапы разработки	5. Состав и содержание работ по созданию системы
8. Порядок контроля и приемки	6. Порядок контроля и приемки системы
	7. Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие
	9.Источники разработки

Опираясь на таблицу, приведенную выше, мы можем выделить основные разделы технического задания:

- Общие сведения о системе (программе);
- Назначение, цели и задачи системы (программы);
- Требования к системе (функциональные требования, пользовательские требования, требования к системе в целом и тд);
- Требования к видам обеспечения;
- Требования к документированию;
- Стадии и этапы разработки;
- Порядок контроля и приемки системы (программы).

Общие сведения

Данный раздел документа Техническое задание должен содержать полное наименование системы и все варианты сокращений, которые будут использованы при разработке документации.

Пример:

«В данном документе создаваемая информационная система называется «Единое окно доступа к образовательным ресурсам», сокращенно ЕО. Систему Единое окно доступа к образовательным ресурсам далее в настоящем документе допускается именовать Единое окно или Система.»

Также сюда следует включить подразделы сообщающие реквизиты организаций участвующих в разработке (Заказчика и Исполнителя).

В подразделе «Основания для разработки» документа Техническое задание перечисляются основные документы, на основании которых выполняются данные работы. Например, для системы, выполняемой по заказу Правительства страны или другого Государственного органа, должны быть указаны законы, указы и постановления Правительства.

Далее следует указать сроки начала и окончания работ и сведения об источнике финансирования. Данная информация может быть указана и вконце технического задания в разделе с указанием стадий и этапов работ.

Неотъемлемой частью документа Техническое задание также должен быть список терминов и сокращений. Термины и сокращения лучше представить в виде таблицы с двумя столбцами «Термин» и «Полная форма».

Термины и сокращения располагаются в алфавитном порядке. В первую очередь принято давать расшифровку русскоязычным терминам и сокращениям, потом англоязычным.

Назначение и цели создания системы

Данный раздел документа Техническое задание должен содержать назначение и цели создания системы.

Пример:

«Информационная система «Единое окно доступа к образовательным ресурсам» предназначена для обеспечения пользователей полной, оперативной и удобной информацией, касающейся системы образования Российской федерации, организаций выполняющих функцию образовательных учреждений.

Основной целью Системы является формирование единой информационной среды и автоматизации бизнес-процессов Образовательных учреждений Российской Федерации.

Создание информационной системы «Единое окно» должно обеспечить:

- *предоставление пользователям широкого спектра информационных ресурсов;*
- *повышение уровня информационной безопасности;*
- *повышение эффективности работы образовательных учреждений и ведомств за счет оптимизации ряда бизнес-процессов;*
- *повышение эффективности процесса взаимодействия информационных систем и сервисов внутри ведомства.*

Создание Системы позволит сократить эксплуатационные затраты в результате повышения эффективности работы ведомства.»

Требования к системе

Данный раздел документа Техническое задание предназначен для описания основных функциональных требований системы. Это самая важная часть технического задания, так как именно она станет основным вашим аргументом при спорах с Заказчиком в процессе сдачи системы в эксплуатацию. Поэтому к его написанию необходимо подойти наиболее тщательно.

В документе Техническое задание должны быть представлены все требования, выявленные на этапе проведения анализа объекта автоматизации. Лучше всего выделить основные бизнес-процессы, которые идолжны быть раскрыты посредством описания функциональных требований.

Пример:

«4.1 Бизнес-процесс «Предоставление информации об образовательных учреждениях Российской Федерации

В данном бизнес-процессе выделяются следующие участники:

Модератор – работник ведомства, входящий в состав обслуживающего персонала Системы, ответственный за корректность предоставляемых данных

Автор – сотрудник образовательного учреждения, ответственный за размещение информации о своей организации.

Пользователь – гражданин, нуждающийся в получении информации о работе образовательных учреждений Российской Федерации.

4.1.1 Регистрация образовательного учреждения в Системе

Регистрация образовательного учреждения Российской Федерации осуществляется ответственным сотрудником учреждения («Постановление Правительства …»).

Процесс регистрации образовательного учреждения включает следующие шаги:

- *Автор создает запись об организации;*
- *Автор заносит данные организации;*
- *Система проверяет наличие лицензии для данной организации*
    - *Если лицензия существует в базе данных, Система отправляет Автору сообщение об успешной регистрации;*
    - *Если лицензия не найдена в базе данных, Система отправляет сообщение Автору об отсутствии лицензии для данной организации.»*

Если позволяет время, информацию, приведенную в данном разделе, следует, более полно раскрыть в приложении к документу Техническое задание. В приложении к техническому заданию можно привести экранную форму и ниже описать все события, которые на ней присутствуют (создание, просмотр, редактирование, удаление и т.п.).

Требования к системе в целом включают раскрытие ее архитектуры с описанием всех подсистем. В данной части Технического задания следует описать требования к интеграции системы с другими продуктами (если таковые имеются). Далее в техническое задание должны быть включены:

- требования к режимам функционирования системы
- показатели назначения
- требования к надежности
- требования к безопасности
- требования к численности и квалификации персонала и режиму его работы
- требования к защите информации
- требования по сохранности информации при авариях
- требования к патентной чистоте
- требования по стандартизации и унификации
- и т.д.

Требованиям к видам обеспечения

В данном разделе документа Техническое задание должны быть представлены требования к математическому, информационному, лингвистическому, программному, техническому и др. видам обеспечения (если таковые имеются).

Требования к документированию

Раздел «Требования к документированию» технического задания включает перечень проектных и эксплуатационных документов, которые должны быть предоставлены заказчику.

Данный раздел технического задания также важен, как и описание функциональных требований, поэтому не следует ограничиваться фразой «Заказчику должна быть предоставлена вся документация согласно ГОСТ 34». Это означает, что вы должны предоставить весь пакет документов включая «Формуляр», «Паспорт» и т.п. Большинство документов из списка, указанного в ГОСТ 34.201-89 не нужны ни вам, ни заказчику, поэтому лучше сразу согласовать список на этапе разработки документа Техническое задание.

Минимальный пакет документов обычно включает:

- Техническое задание;
- Ведомость эскизного (технического) проекта;
- Пояснительная записка к Техническому проекту;
- Описание организации информационной базы;
- Руководство пользователя;
- Руководство администратора;
- Программа и методика испытаний;

- Протокол приемочных испытаний;
- Акт выполненных работ

Перечень документов в техническом задании лучше представить в виде таблицы, где указывается наименование документа и стандарт на основании, которого он должен быть разработан.

Стадии и этапы разработки

В данном разделе документа Техническое задание следует представить информацию обо всех этапах работ, которые должны быть проведены.

Описание этапа должно включать наименование, сроки, описание работ и конечный результат.

Порядок контроля и приемки системы

В данном разделе документа Техническое задание необходимо указать документ, на основании которого должны быть проведены приемо- сдаточные испытания.

При необходимости техническое задание может быть дополнено другими разделами, или сокращено путем удаления нецелесообразных пунктов.

При изменении структуры технического задания, во избежание конфликтных ситуаций, ее необходимо согласовать с заказчиком до разработки документа.

3. IEEE 830-1998 Recommended Practice for Software Requirements Specifications (рекомендуемые методы спецификации требований к ПО).

Данный стандарт описывает рекомендуемые принципы составления спецификации требований к программному обеспечению. Она основана на модели, в которой результат процесса спецификации программного обеспечения является однозначным и полным документом.

Стандарт предполагает следующую структуру спецификаций.

1. ВВЕДЕНИЕ
    1.1. Назначение (указывается назначение ТЗ и ауди тории для которой оно разрабатывается)
    1.2. Область действия (указывается название разра батываемого программного обеспечения, от- мечается, что будет делать программа, какие выгоды

Раздел Специфические требования должен содержать все требования к программному обеспечению на уровне детализации, достаточной, чтобы дать проектировщикам возможность разработать систему, удовлетворяющую этим требованиям, а испытателям - убедиться, что система удовлетворяет этим требованиям. В этом разделе каждое сформулированное требованиедолжно восприниматься пользователями, операторами или другими внешними системами. Эти требования должны включать как минимум описание каждого входного воздействия (стимула) на систему, каждого выходного сигнала (отклика) системы и всех функций, выполняемых системой в ответ на входное воздействие или для поддержания выходного сигнала. Поскольку часто эта часть является самой большой и наиболее важной частью SRS, применяются следующие принципы:

Ниже приведен пример организации структуры специфических требований организованного по функциональной иерархии.

3. СПЕЦИФИЧЕСКИЕ ТРЕБОВАНИЯ

3.1 Требования к внешним интерфейсам

3.1.1 Интерфейсы пользователя

3.1.2 Аппаратные интерфейсы

3.1.3 Интерфейсы программного обеспечения

3.1.4 Интерфейсы связи

3.2 Функциональные требования 3.2.1 Информационные потоки

3.2.1.1 Схема потока данных 1

3.2.1.1.1Информационные объекты

3.2.1.1.2Релевантные потоки

3.2.1.1.3Т опология

3.2.1.1 Схема потока данных 2

3.2.1.2.1Информационные объекты

3.2.1.2.2Релевантные потоки

3.2.1.2.3Топология

3.2.1 .п Схема потока данных *n*

3.2.1. *n.1* Информационные объекты

3.2.1. п..2 Релевантные потоки

3.2.1. п.З Топология

3.2.2 Описания процессов

3.2.2.1 Процесс 1

3.2.2.1.1 Объекты входных данных

3.2.2.1.2Алгоритм или формула процесса

3.2.2.1.3Объекты обрабатываемых данных

3.2.2.2 Процесс 2 3.2.2.2.1Объекты входных данных 3.2.2.2.2Алгоритм или формула процесса 3.2.2.2.3Объекты обрабатываемых данных

3.2.2. *m* Процесс *m*

3.2.2. m..l Объекты входных данных m.1,

3.2.2. т.2 Алгоритм или формула процесса

3.2.1 тЗ Объекты обрабатываемых данных

3.2.2 Спецификации структуры данных

3.2.3.1 Структура 1

3.2.3.1.1Тип записи

3.2.3.1.2Составляющие подя

3.2.3.2 Структура 2

3..2.3.2.1 Тип записи

3.2.3.2.2 Составляющие поля

3.2.3. p Структура*p*

3.2.3. p.1 Тип записи

3.2.3. p.2 Составляющие поля

3.2.3 Словарь данных

3.2.4.1 Элемент данных 1

3.2.4.1.1Имя

3.2.4.1.2Представление

3.2.4.1.3Единицы/Формат

3.2.4.1.4Р азрядно сть/Т очно сть

3.2.4.1.5Диапазон

3.2.4.2 Элемент данных 2

3.2.4.2.1Имя

3.2.4.2.2Представление

3.2.4.2.3Единицы/Формат

3.2.4.2.4Р азрядно сть/Т очно сть

3.2.4.2.5Диапазон

3.2.4. q Элемент данных *q*

3.2.4. q.1 Имя

3.2.4. q.2 Представление

3.2.4. q.3 Единицы/Формат

3.2.4. *q.4* Разрядность/Точность

3.2.4. q.5 Диапазон

3.1 Требования к рабочим характеристикам

3.2 Проектные ограничения

3.3 Атрибуты системы программного обеспечения

3.4 Другие требования

В стандарте приведены несколько примеров организации структуры специфического раздела спецификации требований. Структура этого раздела зависит от типа разрабатываемого программного средства и наиболееточного и полного перечисления всех требований.

4. Задание

Приведенные примеры не являются точными шаблонами длясоставления технического задания. В каждом конкретном случае необходимо придерживаться одного стандарта, допускается не включать разделы, на которые отсутствуют требования. При этом предпочтения при выборе стандарта для разработки технического задания следует отдавать стандарту IEEE 830 так как он является более новым, дает большие свободы

разработчикам с точки зрения структуры документа. В соответствии с выбранной темой и результатами, полученными при выполнении предыдущих работ разработать техническое задание по одной из приведенных форм с уточнением требований по соответствующему стандарту.

Список использованных источников

1. ГОСТ 34.602.89 «Техническое задание на создание автоматизированной системы»
2. ГОСТ 19.201-78 «Техническое задание. Требования к содержанию и оформлению».
3. IEEE 830-1998 «Методика составления спецификаций требований к программному обеспечению».
4. Липаев, В.В. Программная инженерия: электронный учебно-методический комплекс [Электронный ресурс] / В.В. Липаев, Б.М. Позднеев. – М-во образования и науки Российской Федерации, ФГБОУ ВПО Московский гос. технологический ун-т "СТАНКИН", Учебно-методическое объединение высш. учебных заведений Российской Федерации по образованию в области автоматизированного машиностроения, 2012 – 337с.

1. Цель работы

Изучение моделирования систем на логическом уровне с использованием диаграмм классов унифицированного языка моделирования UML.

2. Основные теоретические положения

2.1. СРЕДСТВА ЛОГИЧЕСКОГО МОДЕЛИРОВАНИЯ ЯЗЫКА UML

Для моделирования систем на логическом уровне используются следующие виды диаграмм:

- ***диаграммы классов (class diagrams)***;
- ***диаграммы поведения системы (behavior diagrams)***, которые включают:
  - ***диаграммы взаимодействия (interaction diagrams)*** - для моделирования процесса обмена сообщениями между объектами, к которым относятся:
    - ***диаграммы последовательности (sequence diagrams)***;
    - ***кооперативные диаграммы (collaboration diagrams)***;
  - ***диаграммы состояний (statechart diagrams)*** - для моделирования поведения объектов системы при переходе из одного состояния в другое;
  - ***диаграммы деятельностей (activity diagrams)*** - для моделирования поведения системы в рамках различных вариантов использования, или моделирования деятельностей.

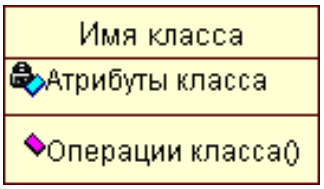
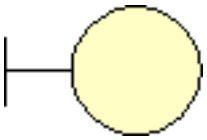
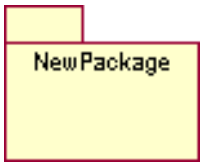
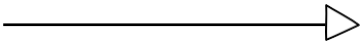
2.2. ДИАГРАММЫ КЛАССОВ

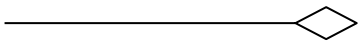
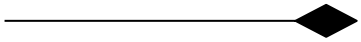
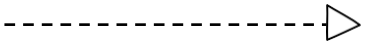
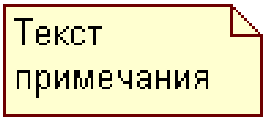
Диаграмма классов ***(class diagrams)*** служит для представления статической структуры модели системы в терминах классов объектно-ориентированного проектирования. Диаграмма классов определяет типы классов системы и различного рода статические связи, которые существуют между ними. На диаграммах классов изображаются также атрибуты классов, операции классов и ограничения, которые накладываются на связи между классами.

Основные элементы диаграммы классов приведены в табл. 1.

Таблица 1

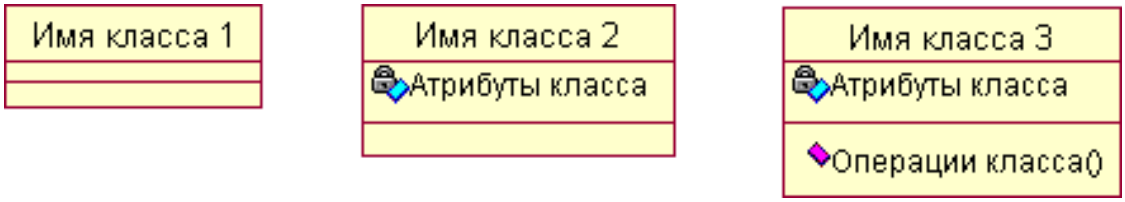
Основные элементы диаграммы классов

Графическое изображение	**Название**
	Класс (class)
	Управляющий класс (control class)
	Класс-сущность (entity class)
	Граничный класс (boundary class)
	Пакет (package)
	Интерфейс (Interface)
	Однонаправленная ассоциация, связь (unidirectional association)
	Обобщение (generalization relationship)

Графическое изображение	Название
	Агрегирование (aggregation relationship)
	Композиция (composition relationship)
	Отношение зависимости (dependency or instantiates)
	Связь класса с ассоциацией (association class)
	Реализация (realize)
	Связь примечания (Anchor Note to Item)
	Примечание (note)
ABC	Надпись (text box)

Класс

Класс (***class***) в языке UML служит для обозначения множества объектов, которые обладают одинаковой структурой, поведением и отношениями с объектами из других классов. Графически класс изображается в виде прямоугольника, который дополнительно может быть разделен горизонтальными линиями на разделы или секции (рис. 1). В этих разделах могут указываться имя класса, атрибуты (переменные) и операции (методы).



а б в

Рис. 1. Графическое изображение класса на диаграмме классов

Обязательным элементов обозначения класса является его имя. На начальных этапах разработки диаграммы отдельные классы могут обозначаться простым прямоугольником с указанием только имени соответствующего класса

(рис. 1, а). По мере проработки отдельных компонентов диаграммы описания классов дополняются атрибутами (рис. 1, б) и операциями (рис. 1, в).

Предполагается, что окончательный вариант диаграммы содержит наиболее полное описание классов, которые состоят из трех разделов или секций. Иногда в обозначениях классов используется дополнительный четвертый раздел, в котором приводится семантическая информация справочного характера или явно указываются исключительные ситуации.

Даже если секция атрибутов и операций является пустой, в обозначении класса она выделяется горизонтальной линией, чтобы сразу отличить класс от других элементов языка UML. Примеры графического изображения классов на диаграмме классов приведены на рис. 2. В первом случае для класса "Прямоугольник" (рис. 2, а) указаны только его атрибуты — точки на координатной плоскости, которые определяют его расположение. Для класса "Окно" (рис. 2, б) указаны только его операции, секция атрибутов оставлена пустой. Для класса "Счет" (рис. 2, в) дополнительно изображена четвертая секция,в которой указано исключение — отказ от обработки просроченной кредитной карточки.

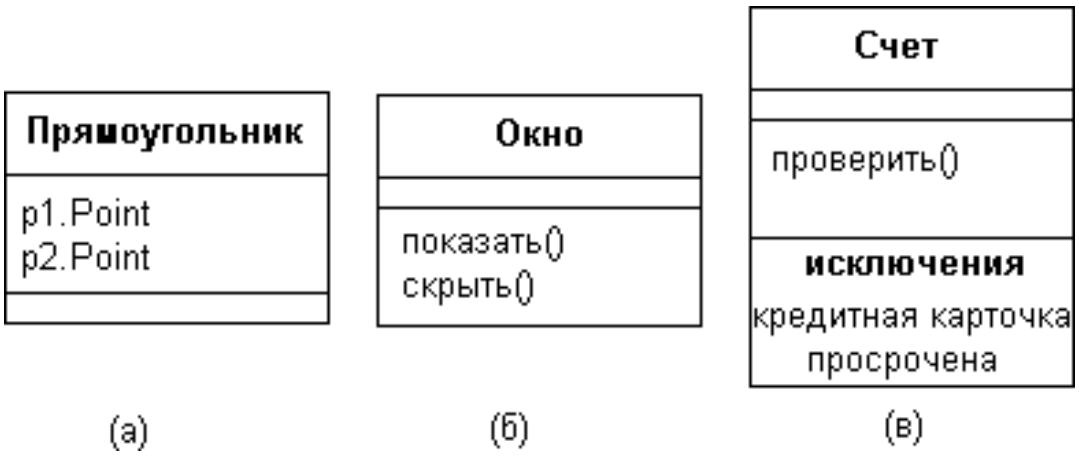


Рис. 2. Примеры графического изображения классов на диаграмме

Стереотипы классов

Стереотипы - это механизм, позволяющий разделять классы на категории. В языке UML основными стереотипами являются: ***Boundary*** (граница), ***Entity*** (сущность) и ***Control*** (управление).

Граничные классы (***boundary classes***) - это классы, которые расположены на границе системы и окружающей среды. Они включают все формы, отчеты, интерфейсы с аппаратурой (такой, как принтеры или сканеры) и интерфейсы с другими системами.

Для того чтобы найти граничные классы, надо исследовать диаграммы вариантов использования. Каждому взаимодействию между действующим лицом и вариантом использования должен соответствовать по крайней мере один граничный класс. Именно такой класс позволяет действующему лицу

взаимодействовать с системой.

Классы-сущности (***entity classes***) отражают основные понятия (абстракции) предметной области и, как правило, содержат хранимую информацию. Обычно для каждого класса-сущности создают таблицу в базе данных.

Управляющие классы (***control classes***) отвечают за координацию действий других классов. Обычно у каждого варианта использования имеется один управляющий класс, контролирующий последовательность событий этого варианта использования. Управляющий класс отвечает за координацию, но сам не несет в себе никакой функциональности - остальные классы не посылают ему большого количества сообщений. Вместо этого он сам посылает множество сообщений. Управляющий класс просто делегирует ответственность другим классам, по этой причине его часто называют классом-менеджером.

В системе могут быть и другие управляющие классы, общие для нескольких вариантов использования. Например класс ***SecurityManager*** (менеджер безопасности), отвечающий за контроль событий, связанных с безопасностью. Класс ***Transaction-Manager*** (менеджер транзакций) занимается координацией сообщений, относящихся к транзакциям с базой данных. Могут быть и другие менеджеры для работы с другими элементами функционирования системы, такими, как разделение ресурсов, распределенная обработка данных или обработка ошибок.

Помимо упомянутых выше стереотипов можно создавать и свои собственные.

Атрибуты

Атрибут (***attribute***) класса - это элемент информации, связанный с классом. Атрибут служит для представления отдельного свойства или признака, который является общим для всех объектов данного класса. Например, у класса ***Company*** (Компания) могут быть атрибуты ***Name*** (Название), ***Address*** (Адрес) и ***NumberOfEmployees*** (Число служащих). Атрибуты записываются во второй сверху секции прямоугольника класса.

Операции

Операция (***operation***) – это некоторый сервис, который предоставляет каждый экземпляр или объект класса по требованию своих клиентов (других объектов, в том числе и экземпляров данного класса). Операции реализуют связанное с классом поведение. Операция включает три части - имя, параметры и тип возвращаемого значения. Параметры - это аргументы, получаемые операцией «на входе». Тип возвращаемого значения относится к результату действия операции. Операции записываются в третьей сверху секции прямоугольника класса.

Механизм пакетов

Пакеты применяют, чтобы сгруппировать классы, обладающие некоторой общностью. Существует несколько наиболее распространенных подходов к

группировке. Во-первых, можно группировать их по стереотипу. В таком случае получается один пакет с классами-сущностями, один с граничными классами,один с управляющими классами и т.д. Этот подход может быть полезен с точки зрения размещения готовой системы, поскольку все находящиеся на клиентских машинах компоненты с граничными классами уже оказываются в одном пакете.

Другой подход заключается в объединении классов по их функциональности. Например, в пакете ***Security*** (безопасность) содержатся все классы, отвечающие за безопасность приложения. В таком случае другие пакеты могут называться ***Employee Maintenance*** (Работа с сотрудниками), ***Reporting*** (Подготовка отчетов) и ***Error Handling*** (Обработка ошибок). Преимущество этого подхода заключается в возможности повторного использования.

Механизм пакетов применим к любым элементам модели, а не только к классам. Если для группировки классов не применять некоторые эвристики, то она становится весьма произвольной. Одна из них, которая в основном используется в UML, - это зависимость. Зависимость между двумя пакетами существует в том случае, если между любыми двумя классами в пакетах существует любая зависимость. Таким образом, ***диаграмма пакетов*** (рис. 13) представляет собой диаграмму, содержащую пакеты классов и зависимости между ними. Строго говоря, пакеты и зависимости являются элементами диаграммы классов, т.е. диаграмма пакетов - это форма диаграммы классов.

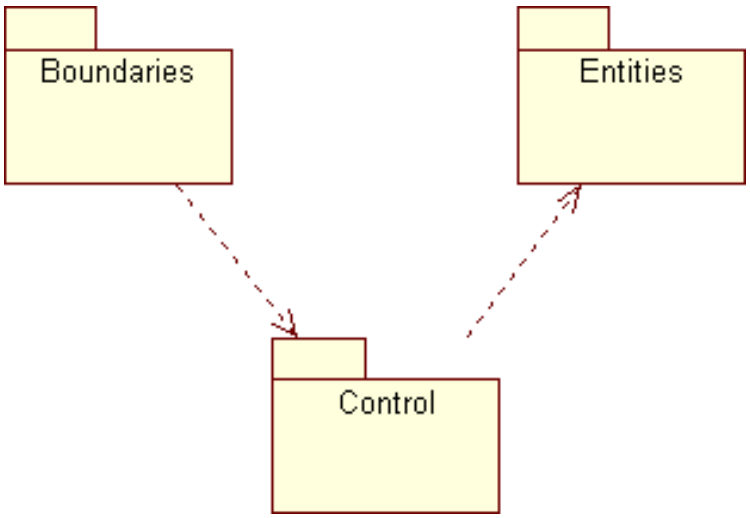


Рис. 13. Диаграмма пакетов

Зависимость между двумя элементами имеет место в том случае, если изменения в определении одного элемента могут повлечь за собой изменения в другом. Что касается классов, то причины для зависимостей могут быть самыми разными: один класс посылает сообщение другому; один класс включает часть данных другого класса; один класс использует другой в качестве параметра операции. Если класс меняет свой интерфейс, то любое сообщение, которое он посылает, может утратить свою силу.

Пакеты не дают ответа на вопрос, каким образом можно уменьшить количество зависимостей в вашей системе, однако они помогают выделить эти

зависимости, а после того поработать над снижением их количества. Диаграммы пакетов можно считать основным средством управления общей структурой системы.

Пакеты являются жизненно необходимым средством для больших проектов. Их следует использовать в тех случаях, когда диаграмма классов, охватывающая всю систему в целом и размещенная на единственном листе бумаги формата А4, становится нечитаемой.

Интерфейсы.

Интерфейс (***interface***) является специальным случаем класса, у которого имеются только операции и отсутствуют атрибуты. Для его изображения используется специальный графический символ — прямоугольник класса со стереотипом <<***interface***>> (рис. 14). При этом секция атрибутов у прямоугольника отсутствует, а указывается только секция операций.

а) б)

Рис. 14. Пример графического изображения интерфейса на диаграмме классов

Пример.

Диаграмма классов для варианта использования «Снять деньги со счета» показана на рис. 15.

На этой диаграмме классов показаны связи между классами, реализующими вариант использования «***Снять деньги со счета***». В этом процессе задействованы четыре класса: ***Card Reader*** (устройство для чтения карточек), ***Account*** (счет), ***ATM Screen*** (экран АТМ) и ***Cash Dispenser*** (кассовый аппарат). Каждый класс на диаграмме выглядит в виде прямоугольника, разделенного натри части. В первой содержится имя класса, во второй - его атрибуты. В последней части содержатся операции класса, отражающие его поведение(действия, выполняемые классом).

Связывающие классы линии отражают взаимодействие между классами.Так, класс ***Account*** связан с классом ***ATM Screen***, потому что они непосредственно сообщаются и взаимодействуют друг с другом. Класс ***Card Reader*** не связан с классом ***Cash Dispenser***, поскольку они не сообщаются друг с другом непосредственно.

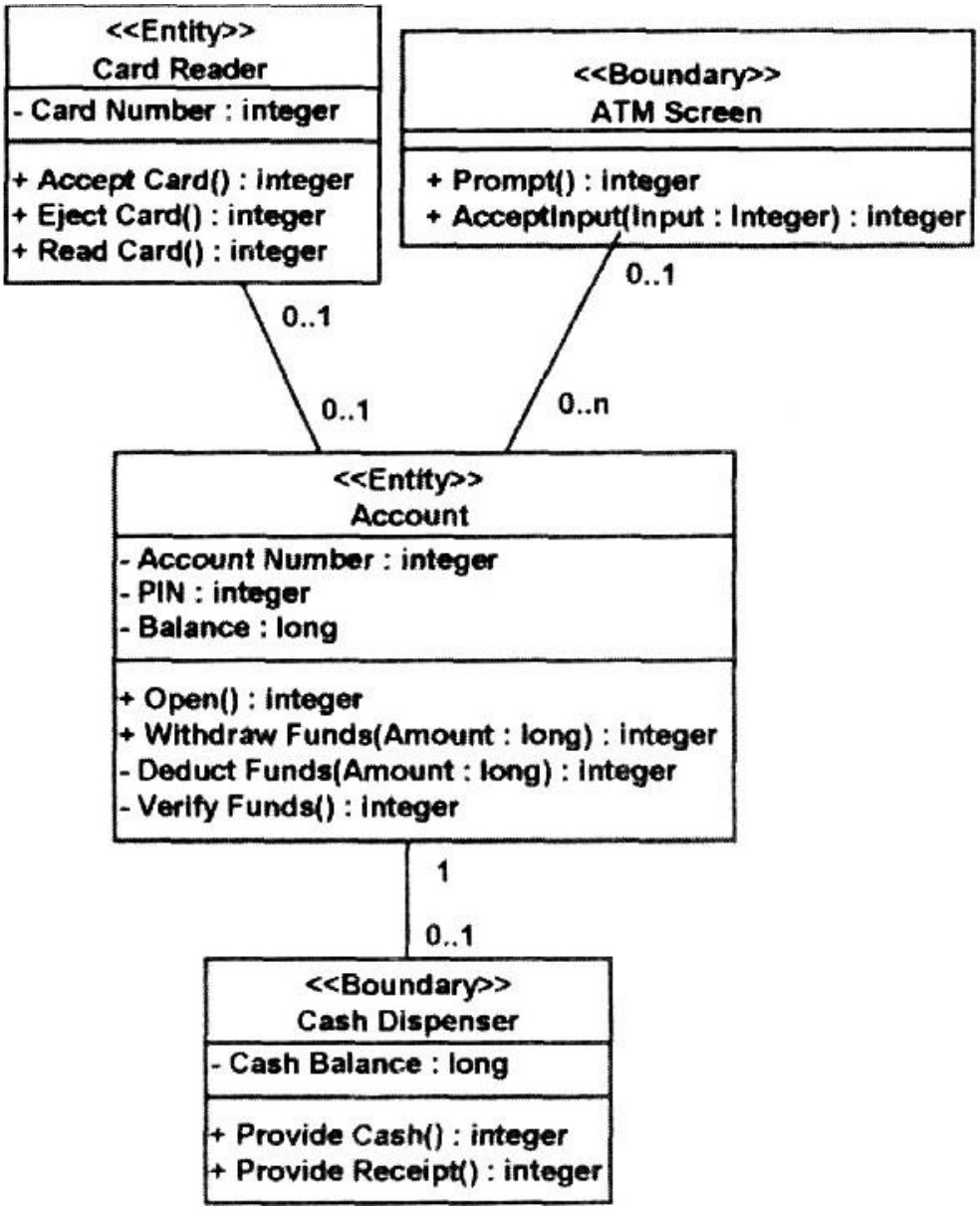


Рис. 15. Диаграмма классов для варианта использования «Снять деньги со счета»

3. ВЫПОЛНЕНИЕ УЧЕБНОГО ПРОЕКТА

3.1. АРХИТЕКТУРНЫЙ АНАЛИЗ СИСТЕМЫ

Принятие соглашений по моделированию включает:

- используемые диаграммы и элементы модели;
- правила их применения;
- соглашения по именованию элементов;
- организацию модели (пакеты).

Пример соглашений моделирования

- Имена вариантов использования должны быть короткими глагольными фразами.
- Для каждого варианта использования должен быть создан пакет ***Use-Case Realization***, включающий:
  - по крайней мере одну реализацию варианта использования;
  - диаграмму «***View Of Participating Classes***» (VOPC).
- Имена классов должны быть существительными, соответствующими по возможности понятиям предметной области.
- Имена классов должны начинаться с заглавной буквы.
- Имена атрибутов и операций должны начинаться со строчной буквы.
- Составные имена должны быть сплошными, без подчеркиваний, каждое отдельное слово должно начинаться с заглавной буквы.

Идентификация ключевых абстракций.

Заключается в предварительном определении классов системы (классов анализа). Источники - знание предметной области, требования к системе, глоссарий. Классы анализа для системы регистрации показаны на рис. 16.

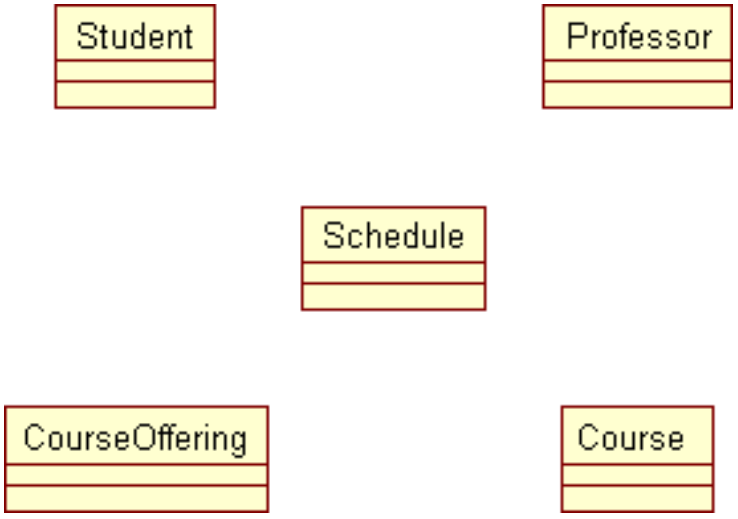


Рис. 16. Классы анализа для системы регистрации

Упражнение 1. Создание структуры модели и классов анализа в соответствии с требованиями архитектурного анализа

Структура логического представления браузера должна иметь следующий вид (рис. 17).

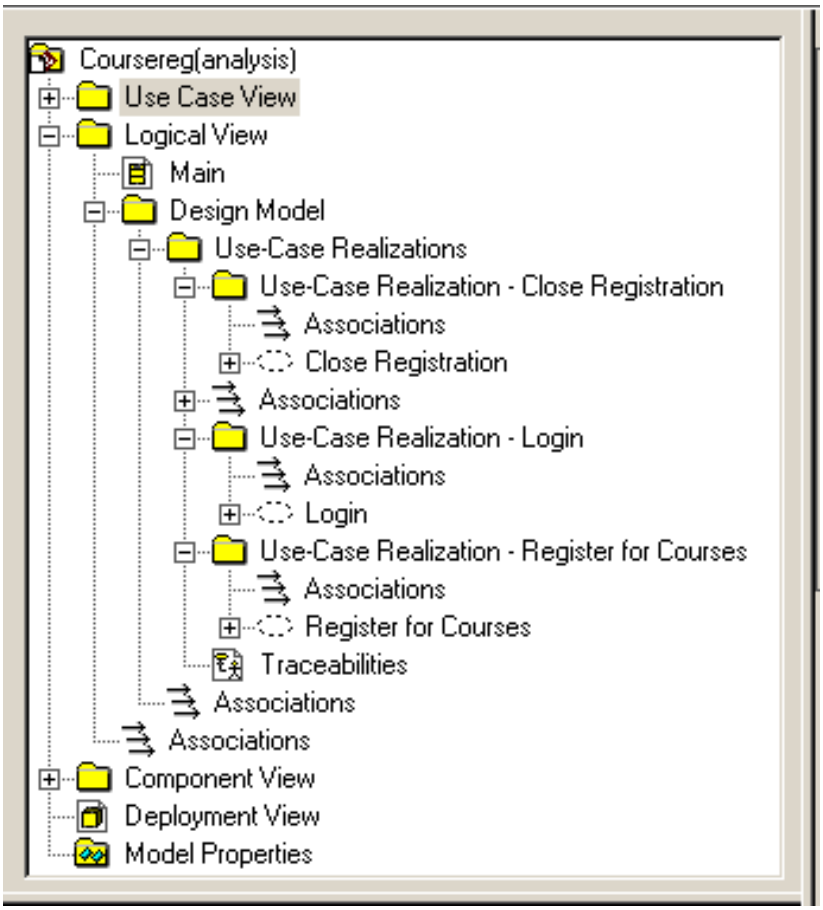


Рис. 17. Структура логического представления браузера

Создание пакетов:

1. Щелкните правой кнопкой мыши по логическому представлению (***Logical View***) браузера.
2. Выберите пункт ***New > Package*** в открывшемся меню.
3. Назовите новый пакет ***Design Model***.

Создание классов анализа и соответствующей диаграммы ***Key Abstractions***:

1. Щелкните правой кнопкой мыши по пакету ***Design Model***.

2. Выберите пункт ***New > Class*** в открывшемся меню. Новый класс под названием ***NewClass*** появится в браузере.

3. Выделите его и введите имя ***Student***.

4. Создайте аналогичным образом классы ***Professor***, ***Schedule***, ***Course*** и ***CourseOffering***.

5. Щелкните правой кнопкой мыши по пакету ***Design Model***.

6. Выберите пункт ***New > Class Diagram*** в открывшемся меню.

7. Назовите новую диаграмму классов ***Key Abstractions***.

8. Чтобы расположить вновь созданные классы на диаграмме классов, откройте ее и перетащите классы на открытую диаграмму мышью. Диаграмма классов должна выглядеть, как на рис. 16.

3.2. ИДЕНТИФИКАЦИЯ КЛАССОВ

Идентификация классов для вариантов использования осуществляется на основе анализа потоков событий варианта использования. В потоках событий варианта использования выявляются классы трех типов:

- ***граничные классы*** (***Boundary***) - служат посредниками при взаимодействии внешних объектов с системой. Как правило, для каждой пары «действующее лицо - вариант использования» определяется один граничный класс. Типы граничных классов: пользовательский интерфейс (обмен информацией с пользователем, без деталей интерфейса - кнопок, списков, окон), системный интерфейс и аппаратный интерфейс (используемые протоколы, без деталей их реализации);

- ***классы-сущности*** (***Entity***) - представляют собой ключевые абстракции (понятия) разрабатываемой системы. Источники выявления классов-сущностей: ключевые абстракции, созданные в процессе архитектурного анализа, глоссарий, описание потоков событий вариантов использования;

- ***управляющие классы*** (***Control***) - обеспечивают координацию поведения объектов в системе. Могут отсутствовать в некоторых вариантах использования, ограничивающихся простыми манипуляциями с хранимыми данными. Как правило, для каждого варианта использования определяется один управляющий класс. Примеры управляющих классов: менеджер транзакций, координатор ресурсов, обработчик ошибок.

Пример набора классов, участвующих в реализации варианта использования ***Register for Courses***, приведен на рис. 18.

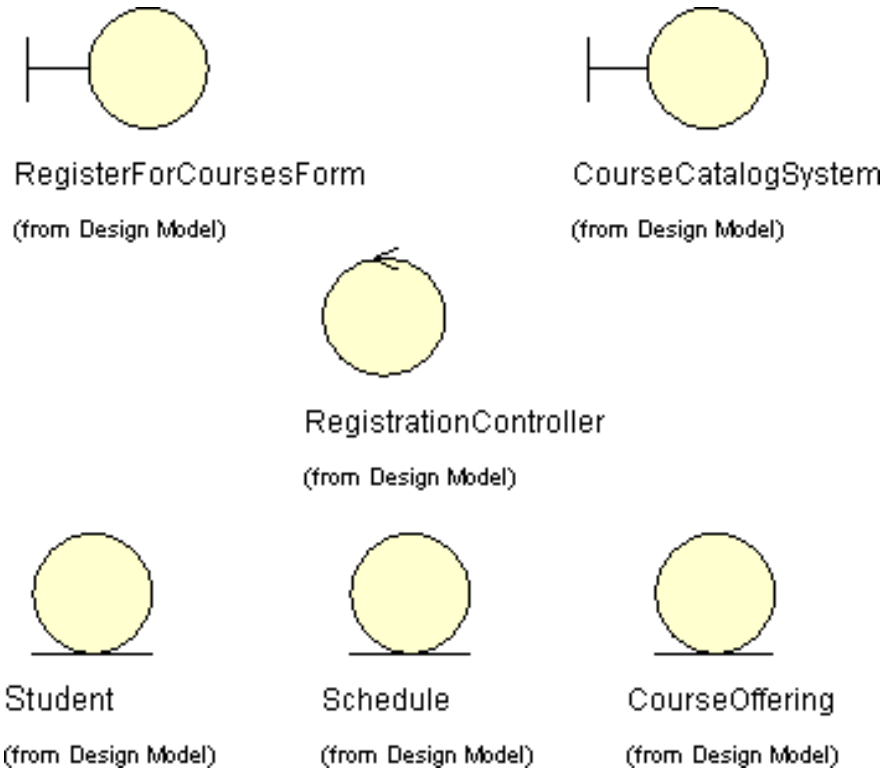


Рис. 18. Классы, участвующие в реализации варианта использования ***Register for Courses***

Упражнение 2. Создание классов, участвующих в реализации варианта использования Register for Courses, и диаграммы классов «View Of Participating Classes» (VOPC)

1. Щелкните правой кнопкой мыши по пакету ***Design Model***.

2. Выберите, пункт ***New*** > ***Class*** в открывшемся меню. Новый класс под названием ***NewClass*** появится в браузере.

3. Выделите его и введите имя ***RegisterForCoursesForm***.

4. Щелкните правой кнопкой мыши по классу ***RegisterForCoursesForm***.

5. Выберите пункт ***Open Specification*** в открывшемся меню.

6. В поле стереотипа выберите ***Boundary*** и нажмите на кнопку ***OK***.

7. Создайте аналогичным образом классы ***CourseCatalogSystem*** со стереотипом ***Boundary*** и ***RegistrationController*** со стереотипом ***Control***.

8. Назначьте классам ***Schedule***, ***CourseOffering*** и ***Student*** стереотип ***Entity***.

9. Щелкните правой кнопкой мыши по кооперации ***Register for Courses*** в пакете ***Use-Case Realization - Register for Courses***.

10. Выберите пункт ***New > Class Diagram*** в открывшемся меню.

11. Назовите новую диаграмму классов ***VOPC*** (***classes only***).

12. Откройте ее и перетащите классы на открытую диаграмму в соответствии с рис. 18.

3.3. СОЗДАНИЕ АТРИБУТОВ И СВЯЗЕЙ МЕЖДУ КЛАССАМИ

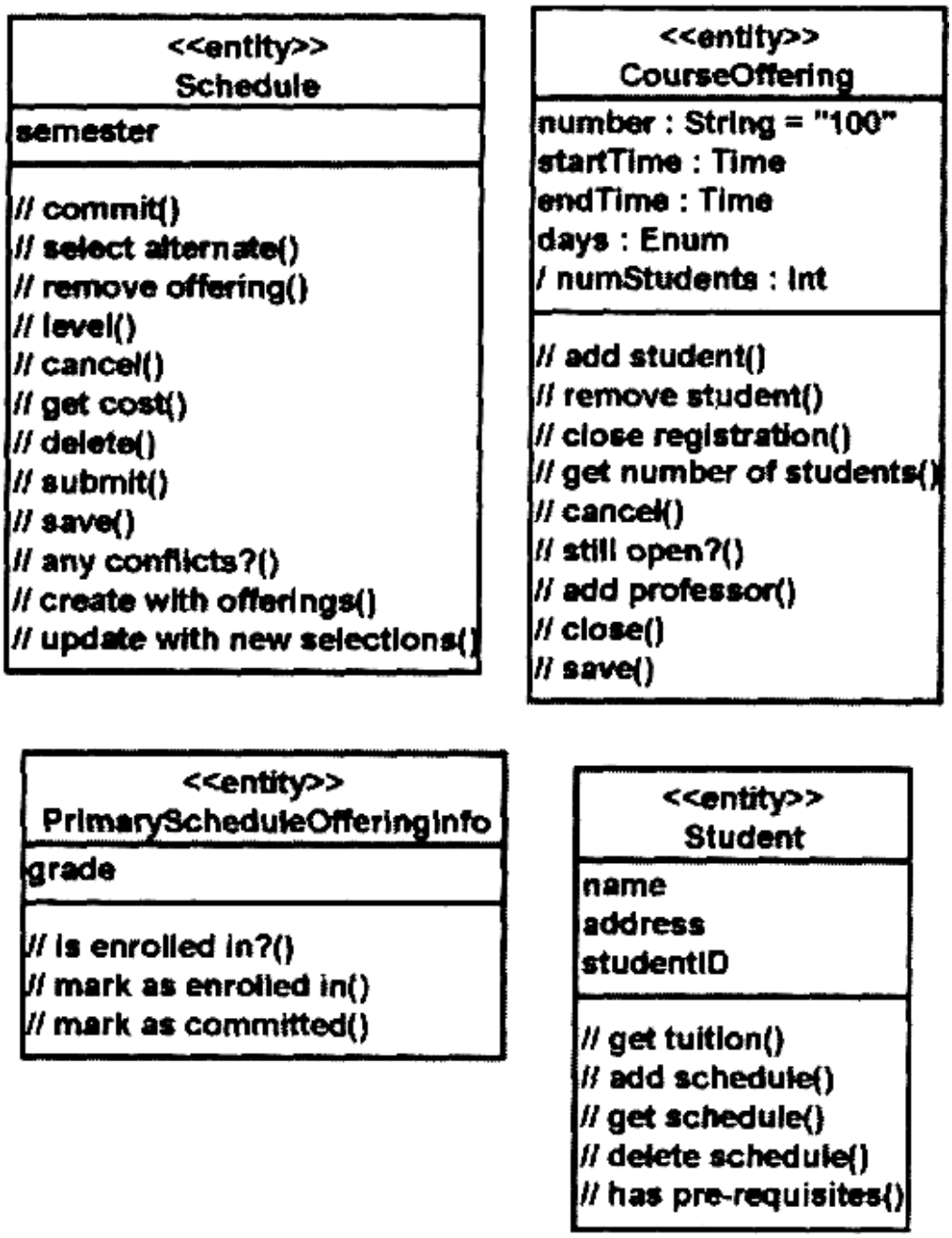
Упражнение 3. Добавление атрибутов к классам

Настройка

1. В меню модели выберите пункт ***Tools > Options***.
2. Перейдите на вкладку ***Diagram***.
3. Убедитесь, что флажок ***Show All Attributes*** включен.
4. Убедитесь, что флажки ***Suppress Attributes*** и ***Suppress Operations*** не включены.

Добавление атрибутов

1. Щелкните правой кнопкой мыши по классу ***Student***.
2. Выберите пункт ***New Attribute*** в открывшемся меню.
3. Введите новый атрибут ***address***.
4. Нажмите клавишу <***Enter***>.
5. Повторите шаги 1-4, добавив атрибуты ***name*** и ***studentID***.
6. Добавьте атрибуты к классам ***CourseOffering***, ***Shedule*** и ***PrimaryScheduleOfferinglnfo***, как показано на рис. 19.


<<entity>>
Schedule
semester
// commit()
// select alternate()
// remove offering()
// level()
// cancel()
// get cost()
// delete()
// submit()
// save()
// any conflicts?()
// create with offerings()
// update with new selections()
<<entity>>
CourseOffering
number : String = "100"
startTime : Time
endTime : Time
days : Enum
/ numStudents : Int
// add student()
// remove student()
// close registration()
// get number of students()
// cancel()
// still open?()
// add professor()
// close()
// save()
<<entity>>
PrimaryScheduleOfferingInfo
grade
// is enrolled in?()
// mark as enrolled in()
// mark as committed()
<<entity>>
Student
name
address
studentID
// get tuition()
// add schedule()
// get schedule()
// delete schedule()
// has pre-requisites()

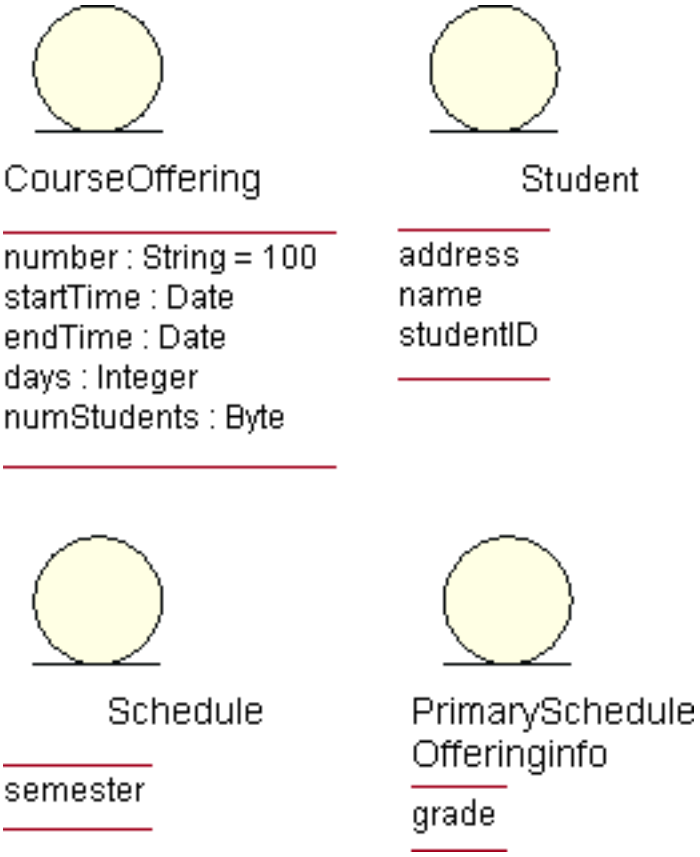


Рис. 19. Классы с атрибутами

Связи между классами (ассоциации) определяются на основе диаграмм взаимодействия. Если два объекта взаимодействуют (обмениваются сообщениями), между ними должна существовать связь (путь взаимодействия). Для ассоциаций задаются множественность и, возможно, направление навигации. Могут использоваться множественные ассоциации, агрегации и классы ассоциаций.

Упражнение 4. Добавление связей

Добавим связи к классам, принимающим участие в варианте использования ***Register for Courses***. Для отображения связей между классами построим три новые диаграммы классов в кооперации ***Register for Courses*** пакета ***Use-CaseRealization - Register for Courses*** (рис. 20- 22).

Добавлены два новых класса - подклассы ***FulltimeStudent*** (Студент очного отделения) и ***ParttimeStudent*** (Студент вечернего отделения).

На данной диаграмме показаны классы ассоциаций, описывающие связи между классами ***Schedule*** и ***CourseOffering***, и добавлен суперкласс ***ScheduleOfferinglnfo***. Данные и операции, содержащиеся в этом классе (***status*** - курс включен в график или отменен), относятся как к основным, так и к альтернативным курсам, в то время как оценка (***grade***) и окончательное включение курса в график могут иметь место только для основных курсов.

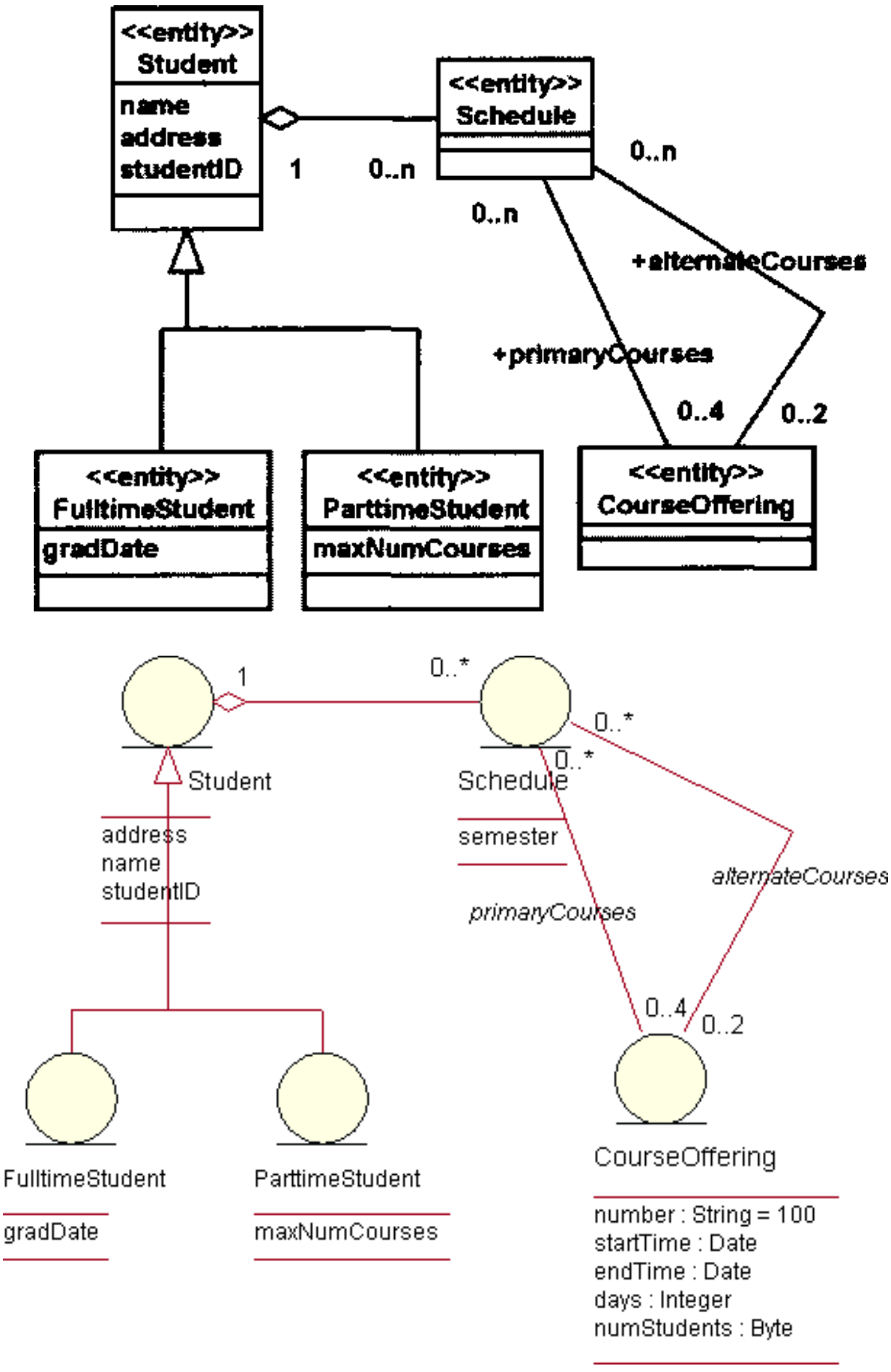


Рис. 20. Диаграмма ***Entity Classes*** (классы-сущности)

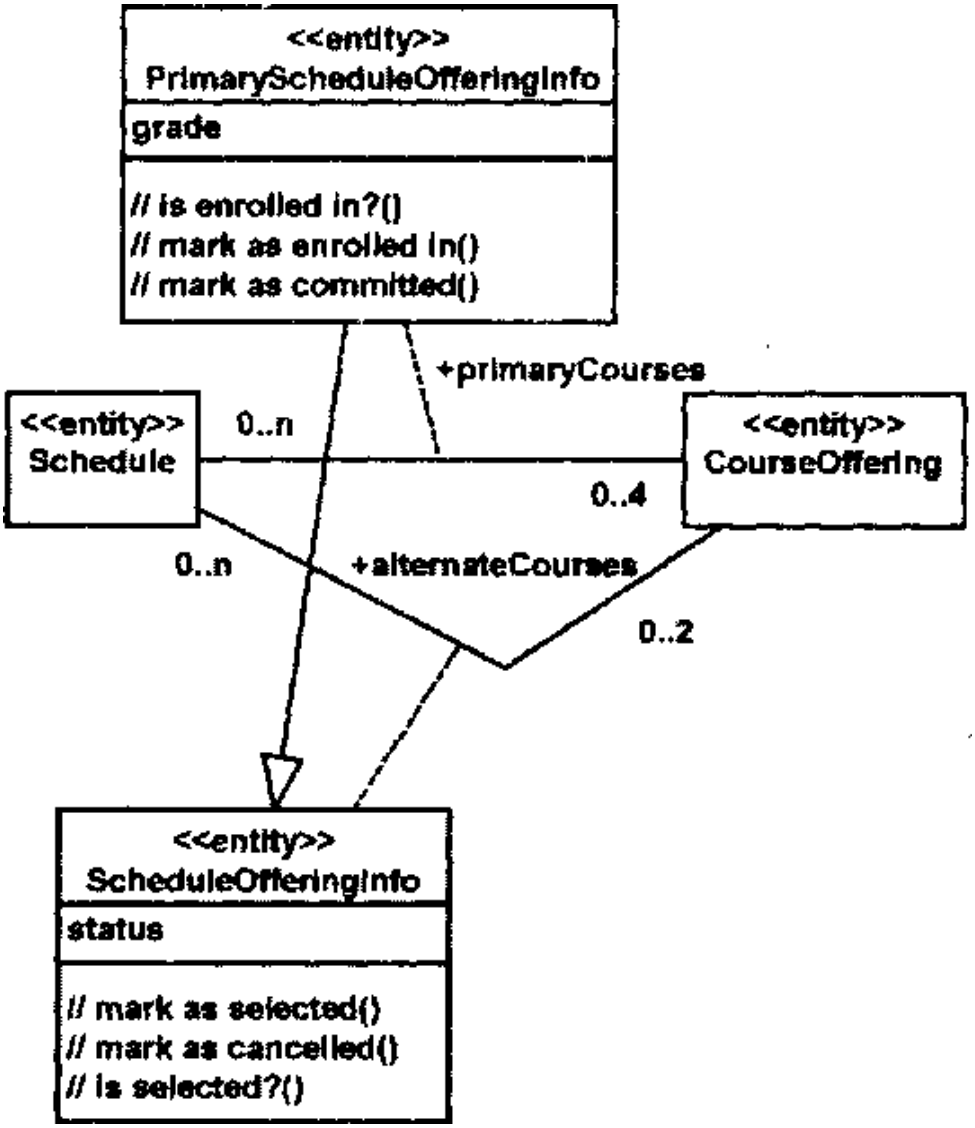


Рис. 21. Диаграмма ***CourseOfferingInfo***

Создание ассоциаций

Ассоциации создают непосредственно на диаграмме классов. Панель инструментов диаграммы классов содержит кнопки для создания как одно-, так и двунаправленных ассоциаций.

Для создания на диаграмме классов ассоциации сделайте следующее:

1. Нажмите на панели инструментов кнопку ***Association***.

2. Проведите мышью линию ассоциации от одного класса к другому.

С целью задать возможности навигации по ассоциации (направления ассоциации) необходимо выполнить следующие действия:

1. Щелкните правой кнопкой мыши по связи с того конца, на которомхотите показать стрелку.

2. Выберите пункт ***Navigable*** в открывшемся меню.

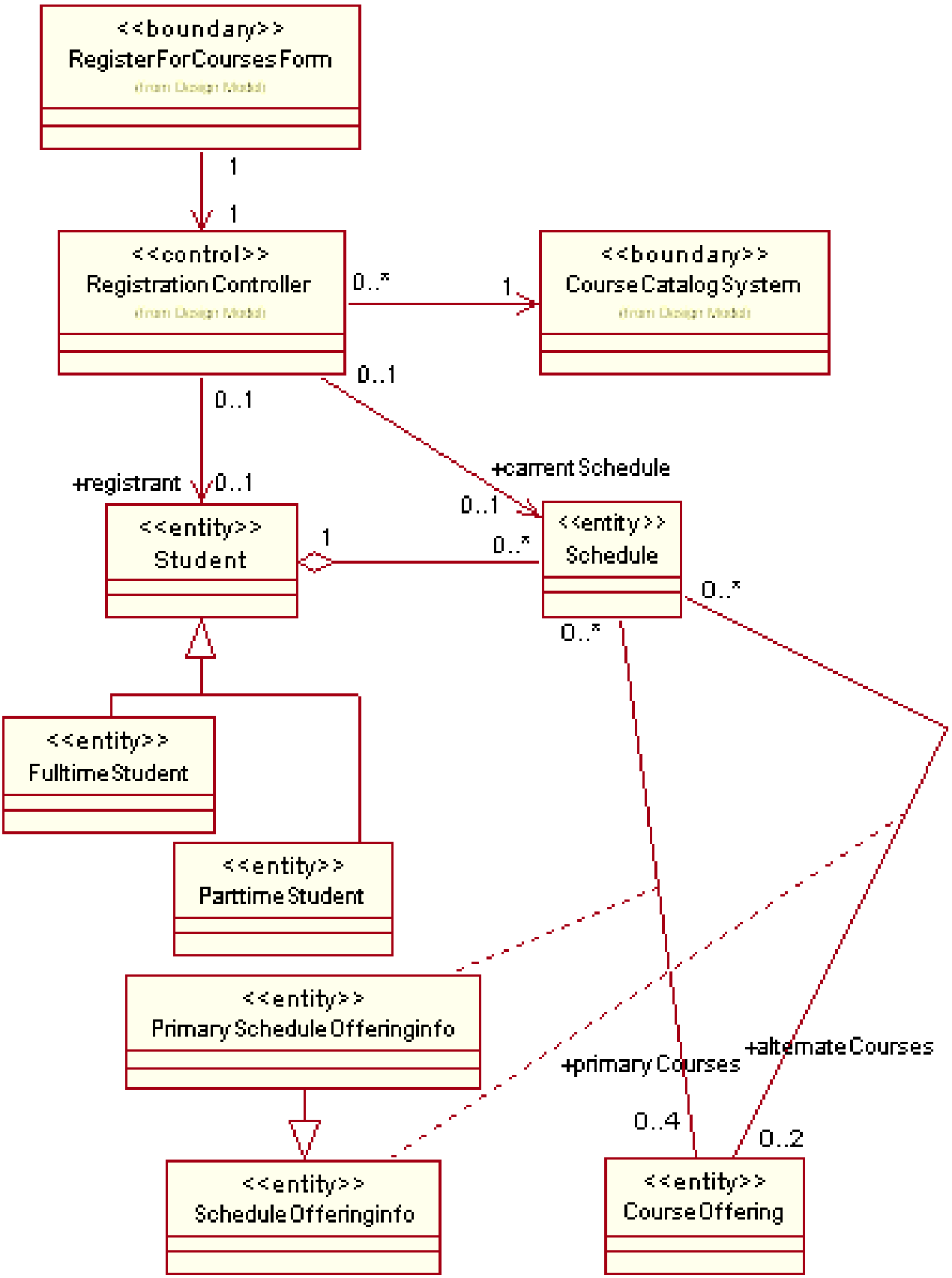


Рис. 22. Полная диаграмма классов ***VOPC*** (без атрибутов и операций)

Для того чтобы создать рефлексивную ассоциацию:
1. На панели инструментов диаграммы нажмите кнопку ***Association***.

2. Проведите линию ассоциации от класса до какого-нибудь места вне класса.

3. Отпустите кнопку мыши.

4. Проведите линию ассоциации назад к классу.

Создание агрегаций

1. Нажмите кнопку ***Aggregation*** панели инструментов.

2. Проведите линию агрегации от класса-части к целому. Для того чтобы поместить на диаграмму классов рефлексивную агрегацию:

1. На панели инструментов диаграммы нажмите кнопку ***Aggregation***.

2. Проведите линию агрегации от класса до какого-нибудь места вне класса.

3. Отпустите кнопку мыши.

4. Проведите линию агрегации назад к классу.

Создание обобщений

При создании обобщения может потребоваться перенести некоторые атрибуты или операции из одного класса в другой. Если, например, понадобится перенести их из подкласса в суперкласс ***Employee***, в браузере для этого достаточно просто перетащить атрибуты или операции из одного класса в другой. Не забудьте удалить другую копию атрибута из второго подкласса, если он имеется.

Чтобы поместить обобщение на диаграмму классов:

1. Нажмите кнопку ***Generalization*** панели инструментов.

2. Проведите линию обобщения от подкласса к суперклассу.

Спецификации связей

Спецификации связей касаются имен ассоциаций, ролевых имен, множественности и классов ассоциаций. Для того чтобы задать множественность связи:

1. Щелкните правой кнопкой мыши на одном конце связи.

2. Выберите пункт ***Multiplicity*** в открывшемся меню.

3. Укажите нужную множественность.

4. Повторите то же самое для другого конца связи.

Для того чтобы задать имя связи:

1. Выделите нужную связь.

2. Введите ее имя.

Для того чтобы задать связи ролевое имя:

1. Щелкните правой кнопкой мыши на ассоциации с нужного конца.

2. Выберите пункт ***role Name*** в открывшемся меню.

3. Введите ролевое имя.

Для того чтобы задать элемент связи (класс ассоциаций):

1. Откройте окно спецификации требуемой связи.

2. Перейдите на вкладку ***Detail***.

3. Задайте элемент связи в поле ***Link Element***.

Рассмотрим пример ИС «Деканат».

Выделим сущности (классы-сущности). Определим связи, типы связей, их имена (рис. 23).

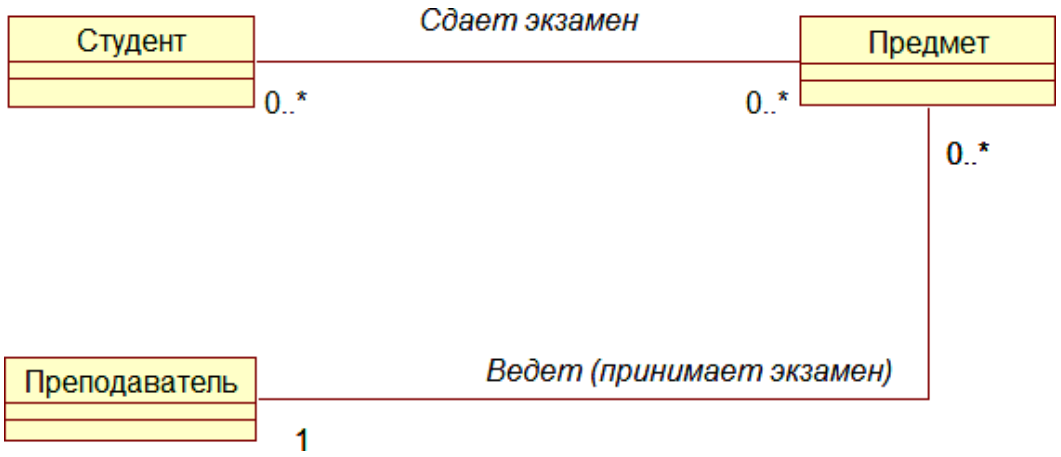


Рис. 23. Инфологическая модель (диаграмма классов ***Key Abstractions***) – первый шаг

Определяем атрибуты сущностей и связей. Поскольку связь СДАЕТ ЭКЗАМЕН имеет два атрибута необходимо ввести зависимую сущность с таким же именем (рис.24).

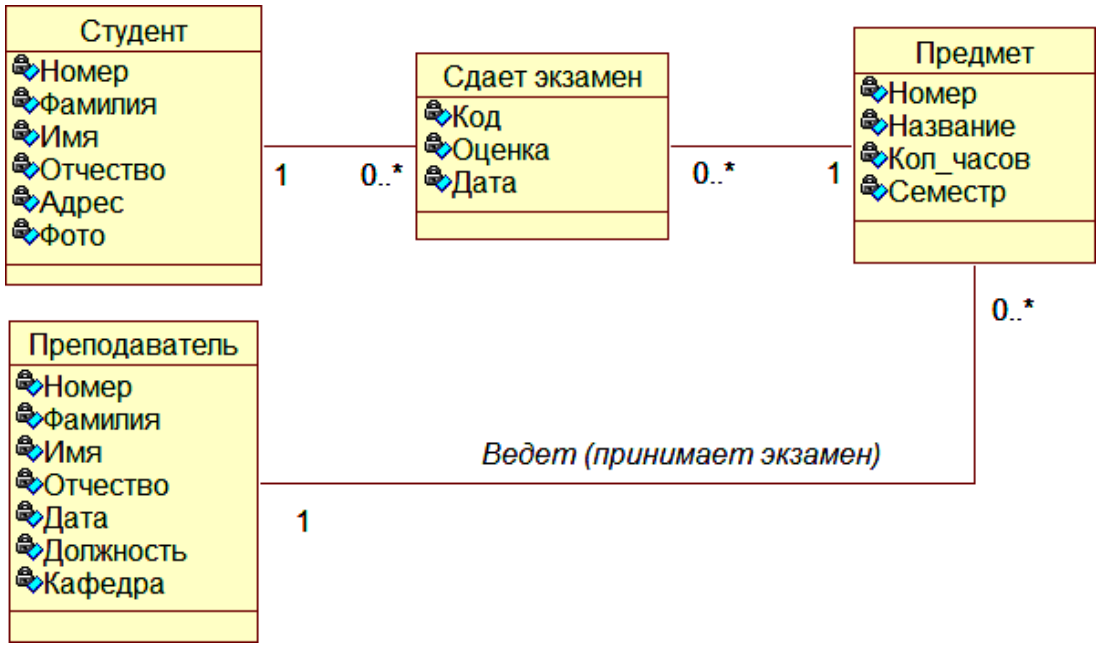


Рис. 24. Инфологическая модель (диаграмма классов ***Key Abstractions***) – второй шаг

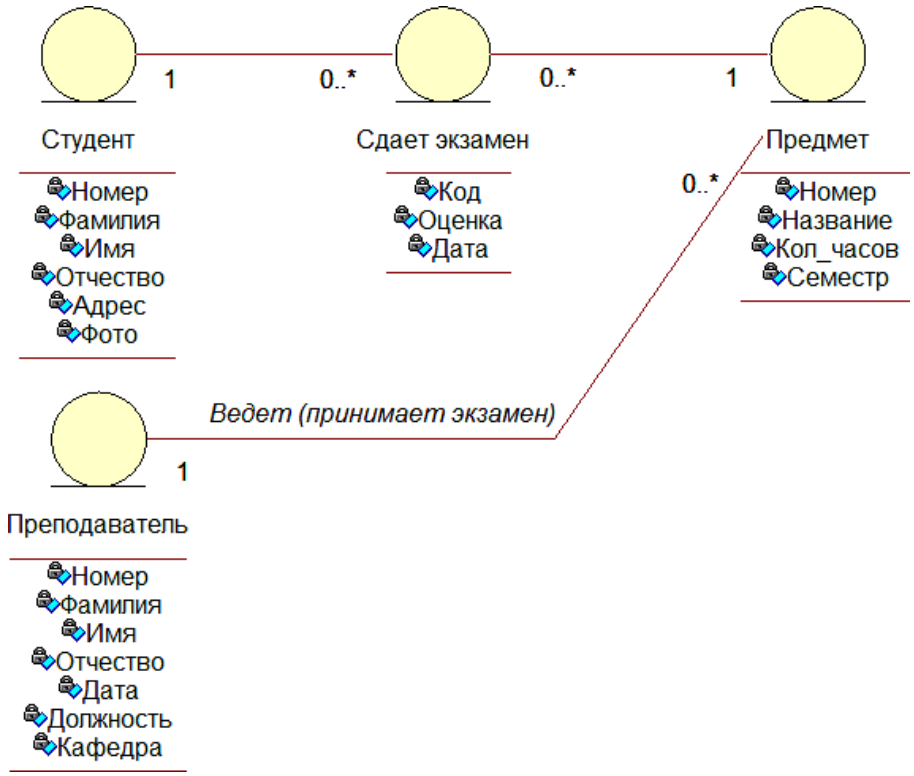


Рис. 25. Инфологическая модель (диаграмма классов ***Key Abstractions***) – другое графическое представление (отображение стереотипа)

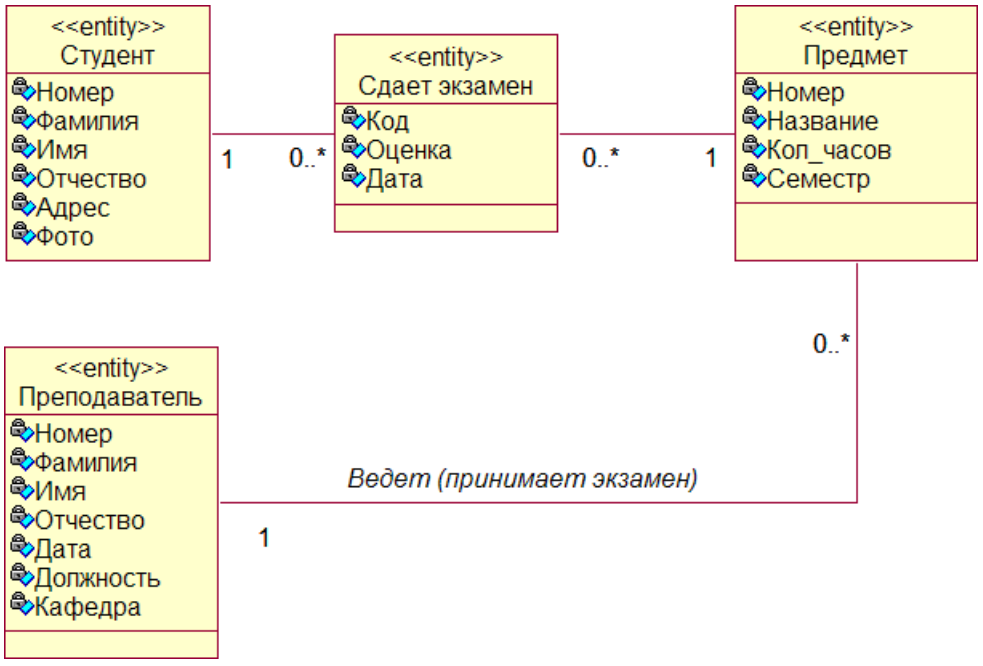


Рис. 26. Инфологическая модель (диаграмма классов ***Key Abstractions***) – другое графическое представление (отображение стереотипа)

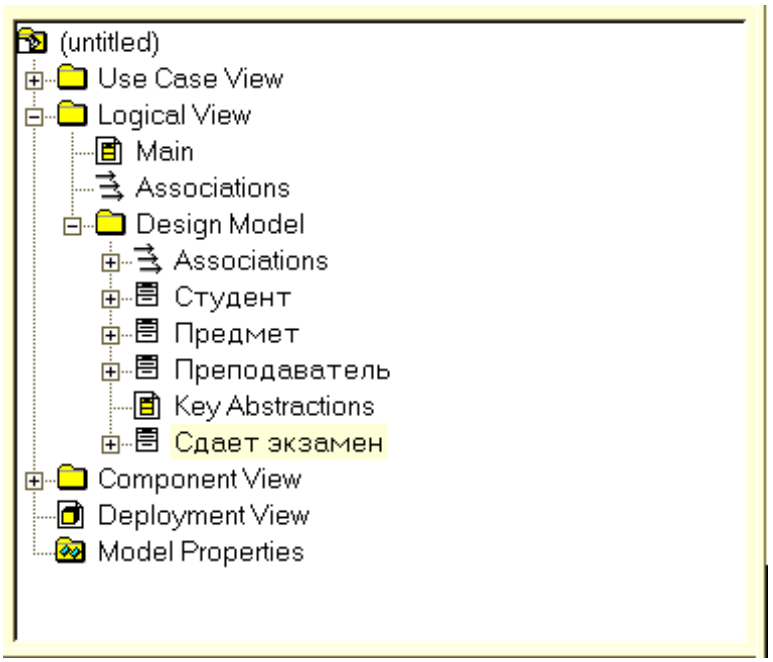


Рис.27. Структура логического представления браузера

4. Выполнение работы

1. Изучить раздел 1 и 2.

2. Изучить раздел 3. Выполнить упражнения 1 – 4.
3. Разработать инфологическую модель системы в виде диаграммы классов в соответствии с вариантом задания.
4. Создать разработанную в п. 3 модель в среде Rational Rose.

5. Содержание отчета

В качестве отчета о выполненной работе предъявите преподавателю:

1. на экране в среде Rational Rose: упражнения 1 – 2 (раздел 3) и инфологическую модель в виде диаграммы классов по индивидуальному заданию;
2. отчет в печатном виде содержащий:
   - задание,
   - диаграмму вариантов использования,
   - диаграмму классов.

6. Контрольные вопросы

1. Каково назначение диаграмм классов?
2. Для чего используется диаграмма классов на стадии анализа?
3. Для чего используется диаграмма классов на стадии проектирования?
4. Назовите основные компоненты диаграмм классов.
5. Назовите основные типы статических связей между классами.

6. Что представляет собой ассоциация?
7. В чем смысл множественности ассоциаций?;
8. В чем отличие атрибутов от ассоциаций?
9. Что такое признак видимости?
10. Что представляет собой операция класса?
11. В чем смысл обобщения?
12. Каково назначение ограничений на диаграммах классов?

7. Библиографический список

1. Вендров А.М. Проектирование программного обеспечения экономических информационных систем: Учебник. - М.: Финансы и статистика, 2002. - 352 с.
2. Вендров А.М. Практикум по проектированию программного обеспечения экономических информационных систем: Учеб. пособие. - М.: Финансы и статистика, 2002. - 192 с.
3. CASE-технологии: Практикум / Федотова Д.Э., Семенов Ю.Д., Чижик К.Н. – М.: Горячая линия – Телеком, 2003. 160 с.

1. Цель работы

Изучение моделирования систем на логическом уровне с использованием диаграмм классов унифицированного языка моделирования UML.

2. Основные теоретические положения

2.1.Технология моделирования поведения системы на основе диаграмм взаимодействия (interaction diagrams) UML

Взаимодействие между объектами в системе представляются *диаграммами взаимодействия (interaction diagrams)*. Диаграммы взаимодействия подразделяются на два основных типа диаграмм: *диаграммы последовательности (sequence diagrams)* и *диаграммы деятельности (activity)*.

Как правило, диаграмма взаимодействия используется для описания поведения в рамках одного варианта использования. На такой диаграмме изображается ряд объектов и те сообщения, которыми они обмениваются в рамках этого варианта использования.

Диаграммы последовательности несут в себе одну информацию, но выраженную разными способами. Диаграммы последовательности показывают взаимодействие объектов во времени и отражают последовательность происходящих событий.

1. Диаграммы последовательности (sequence diagrams)

Диаграммы последовательности имеют две размерности: вертикальная представляет время, горизонтальная - различные объекты. Оси могут меняться местами, так что ось времени может располагаться горизонтально, слева направо, а список объектов располагаться вертикально.

Объект на диаграмме изображается в виде прямоугольника на вершине вертикальной пунктирной линии, называемой *линией жизни объекта (lifeline)* (рис.2.20). Она представляет собой фрагмент жизненного цикла объекта в процессе взаимодействия. Если объект создается или уничтожаетсяна отрезке времени, представленном на диаграмме, то его линия жизни начинается и заканчивается в соответствующих точках, в противном случае линия жизни объекта проводится от начала до конца диаграммы. Символ объекта рисуется в начале его линии жизни; если объект создается не в начале диаграммы, то сообщение о создании объекта рисуется со стрелкой, проведенной к символу объекта. Если объект уничтожается не в конце диаграммы, то момент его уничтожения помечается большим крестиком "X". При сообщении, вызывающем уничтожение объекта (или самоуничтожение), в конце возвращается сообщение об уничтожении объекта. Линия жизни может разветвляться в две (и более) параллельные линии, показываемые условно. Каждая ответвляющаяся линия соответствует переходу в потоке сообщений. Линии жизни могут объединяться в некоторой последующей отметке.

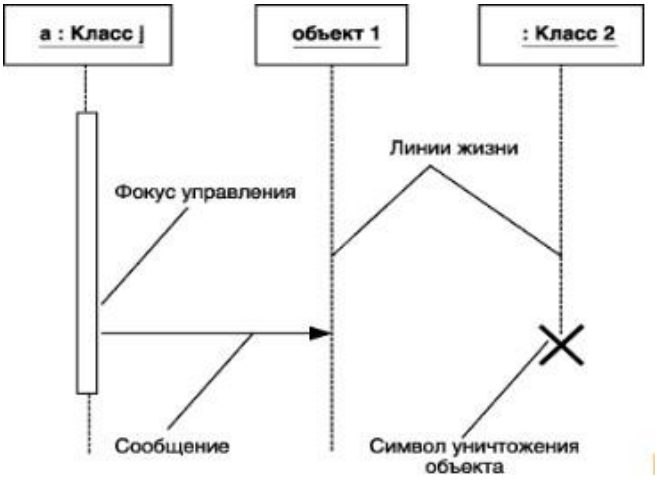


Рисунок 1 –Диаграмма последовательности

Сообщения (message) связывают объекты между собой и передают информацию о выполняемом действии.

Сообщения могут быть следующих типов:

Сообщения могут быть следующих видов:

→ – ***синхронное*** сообщение (англ. synchronous message). Клиент посылает сообщение серверу и ждет, пока тот примет и обработаетсообщение. Как правило, один объект передает синхронное сообщение второму, второй – третьему и т.д., образуя вложенный поток сообщений. В любом случае клиент, инициирующий поток сообщений, должен дождаться его завершения, т.е. возврата управления. Это самый распространенный тип сообщений;

→ – ***асинхронное*** сообщение (англ. asynchronous message). Клиент посылает сообщение серверу и, не дожидаясь ответа, продолжает выполнять следующие операции;

⇢ – ***возвращающее*** сообщение (англ. reply message), обозначающее возврат значения или управления от сервера обратно клиенту.

Сообщения, получаемые от внешнего источника (англ. found message) и передаваемые внешнему приемнику (англ. lost message), должны начинаться и заканчиваться закрашенным кружком.

Диаграммы последовательности полезны для представления параллельных процессов. Для этого в диаграммах последовательности вводятся *активации (activation).*

Активация Ф***окус управления*** (англ. focus of control) показывает период времени, в течение которого объект выполняет действия непосредственно или через зависимую процедуру. Пример диаграммы последовательбности на рис.2.

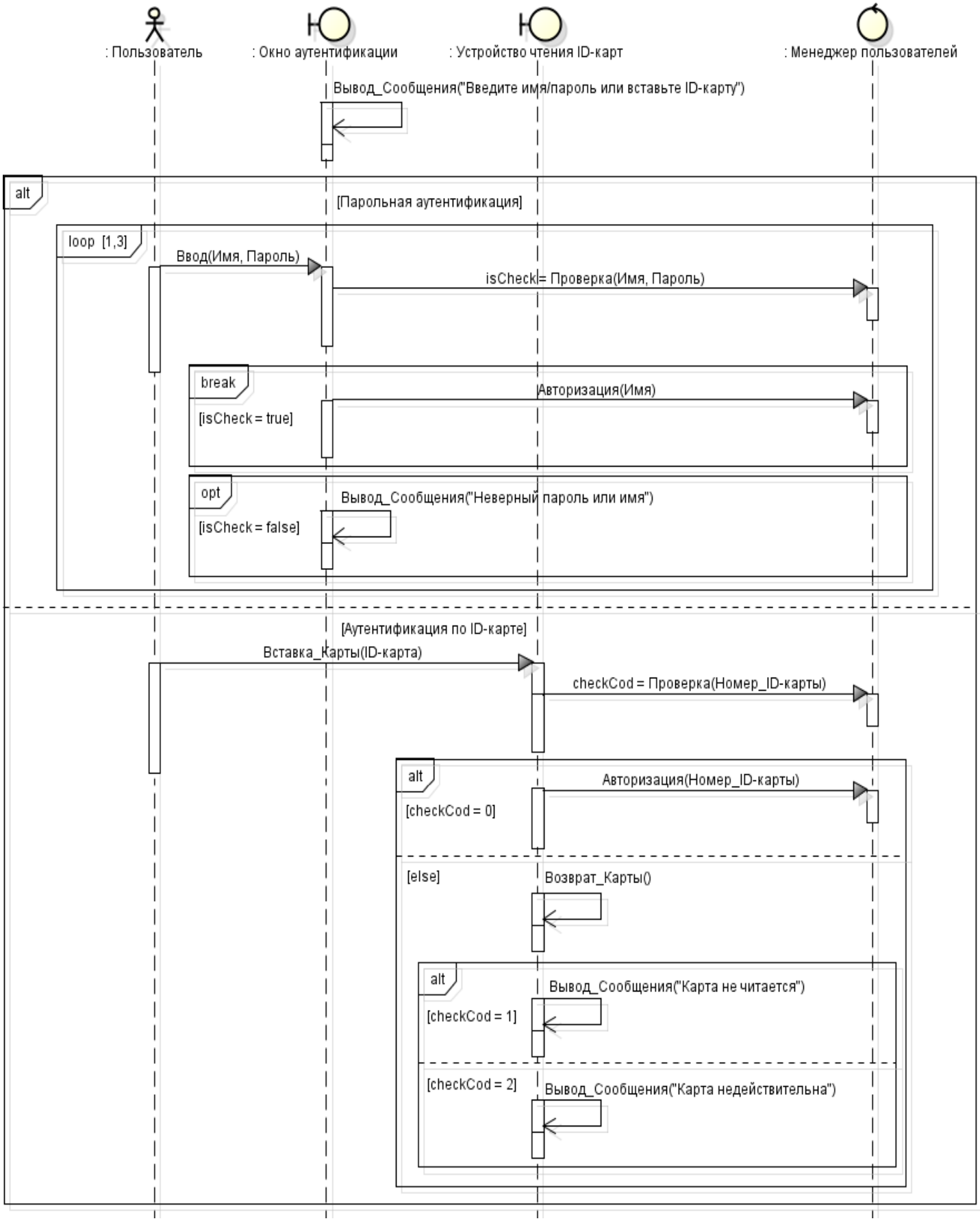


Рисунок 2 – Диаграмма последовательности

2. Построение диаграммы активности (Activity)

Диаграмма активности – позволяет моделировать **жизненный цикл объекта** в **виде** переходами из одного состояния (деятельности) в другое, т.е. отображают алгоритмы по преобразованию классов. Этот тип диаграмм позволяет проектировать алгоритмы поведения объектов с применением обозначений:

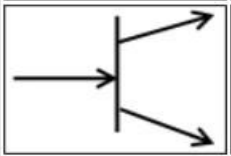
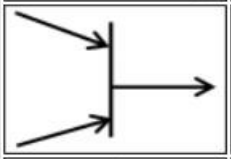
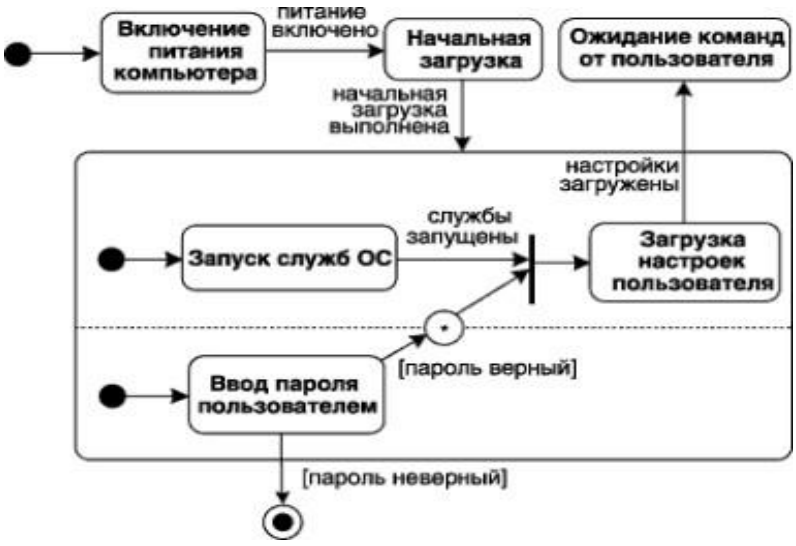
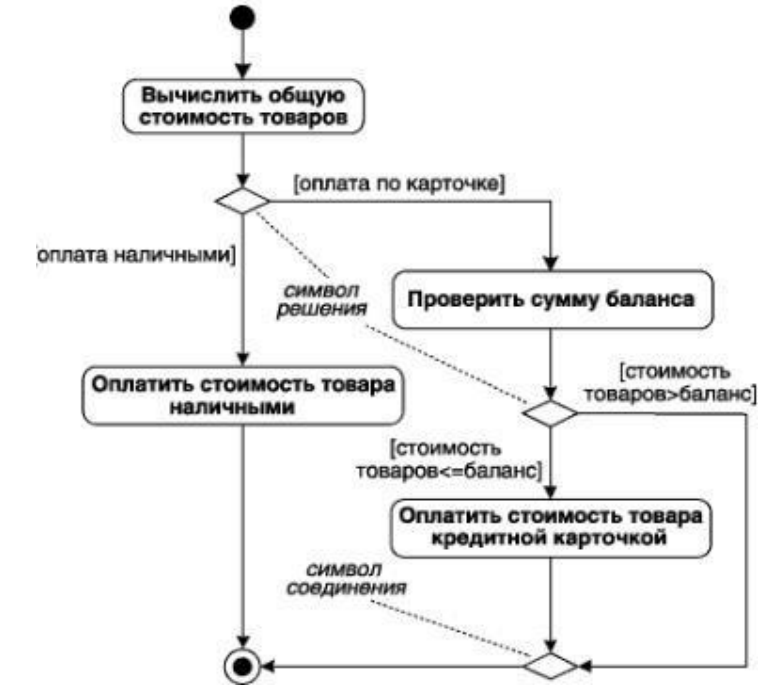
Обозначение	Наименование	
	на диаграмме автоматов	на диаграмме деятельности
	Начальное псевдосостояние (англ. initial pseudostate)	Начальный узел (англ. initial node)
	Конечное состояние (англ. final state)	Завершение деятельности (англ. activity final)
	Точка выхода (англ. exit point pseudostate)	Завершение потока (англ. flow final)
	Ветвление (англ. fork pseudostate)	Ветвление (англ. fork node)
	Соединение (англ. join pseudostate)	Соединение (англ. join node)
	Выбор (англ. choice pseudostate)	Слияние / решение (англ. merge / decision node)

Диаграмма активности позволяет разделить функции по исполнителям с использованием разделов диаграмм. Разделы группируют действия относительно какой-либо характеристики, например:





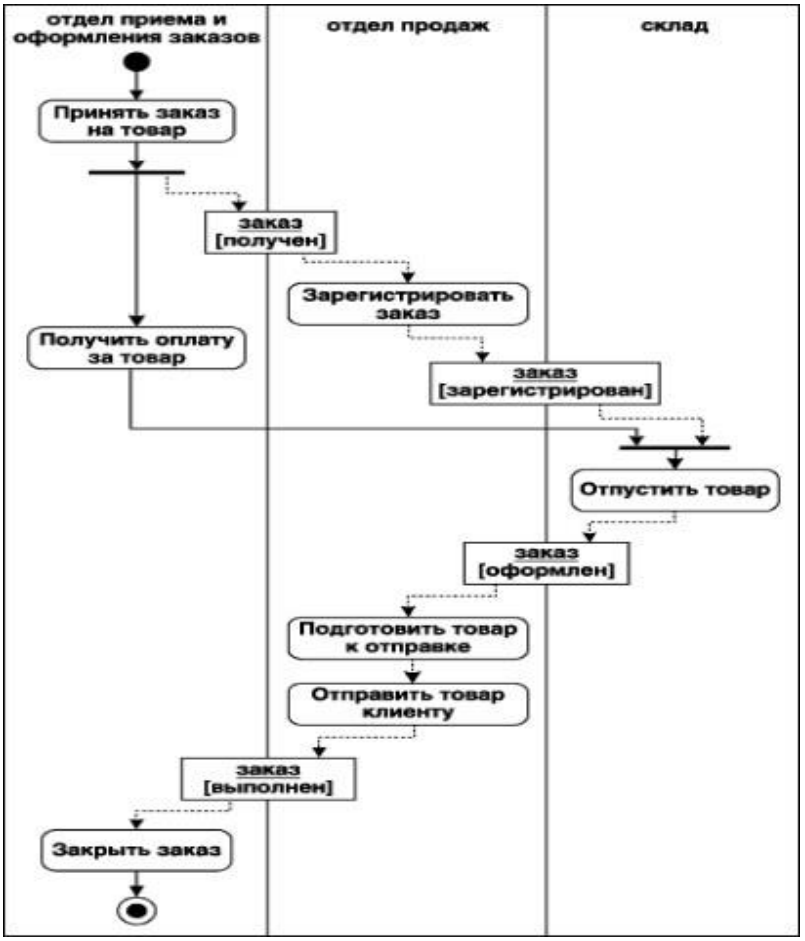


Рисунок 3 – Диаграммы деятельности (активности)

3. Кооперативные диаграммы (collaboration diagrams)

Кооперативные диаграммы (collaboration diagrams) предоставляют возможность пространственно располагать объекты. В отличие от диаграмм последовательности, на кооперативных диаграммах экземпляры объектов показываются в виде пиктограмм. На диаграмме отображаются лишь те объекты, что прямо или косвенно участвуют в выполнении данного варианта использования. Так же как на диаграмме последовательности, линии со стрелкой на конце обозначают сообщения, обмен которыми осуществляется в рамках данного варианта использования. Их временная последовательность, однако, указывается путем нумерации сообщений.

Линия со стрелкой проводится около линии, соединяющей объекты и указывает в направлении объекта, которому посылается сообщение. Для пометки различных сообщений могут использоваться следующие типы стрелок:

- *Линия с заполненной стрелкой*. Обозначает вызов процедуры. Может использоваться также между параллельно работающими активными объектами для посылки сигналов и ожиданий.
- *Линия с половинкой стрелки.* Асинхронный поток управления. Используется для явного указания на асинхронный обмен сообщениями между двумя объектами.
- *Другие разновидности.* Могут представлять другие разновидности управления, например, *"balking"* или *"timeout"*, но они обычно воспринимаются как дополнительные возможности UML.

Сообщения на кооперативной диаграмме помечаются номерами. Нумерация сообщений делает восприятие их последовательности более трудным, чем в случае расположения линий на странице сверху вниз. Принято применять вложенную систему нумерации, так как это позволяет понять какая операция вызывает какую операцию, хотя при этом может быть труднее разглядеть их общую последовательность.

Внутренние сообщения о выполнении операции нумеруются, начиная с 1. В последовательности сообщений между параллельными объектами нумерация сообщений относится к одному уровню (нет вложенности). На кооперативной диаграмме сообщение можно снабдить такой же управляющей информацией, что и на диаграмме последовательности.

Пиктограмма объекта на кооперативной диаграмме помечается строкой имени, имеющей вид:

<ИмяОбъекта : Имя Класса>

где имя объекта, либо имя класса могут отсутствовать. Обратите внимание, что если имя объекта отсутствует, то перед именем класса для ясности сохраняется двоеточие.

Вызов взаимодействия на диаграмме может быть представлен символом действующего лица (рис.3).

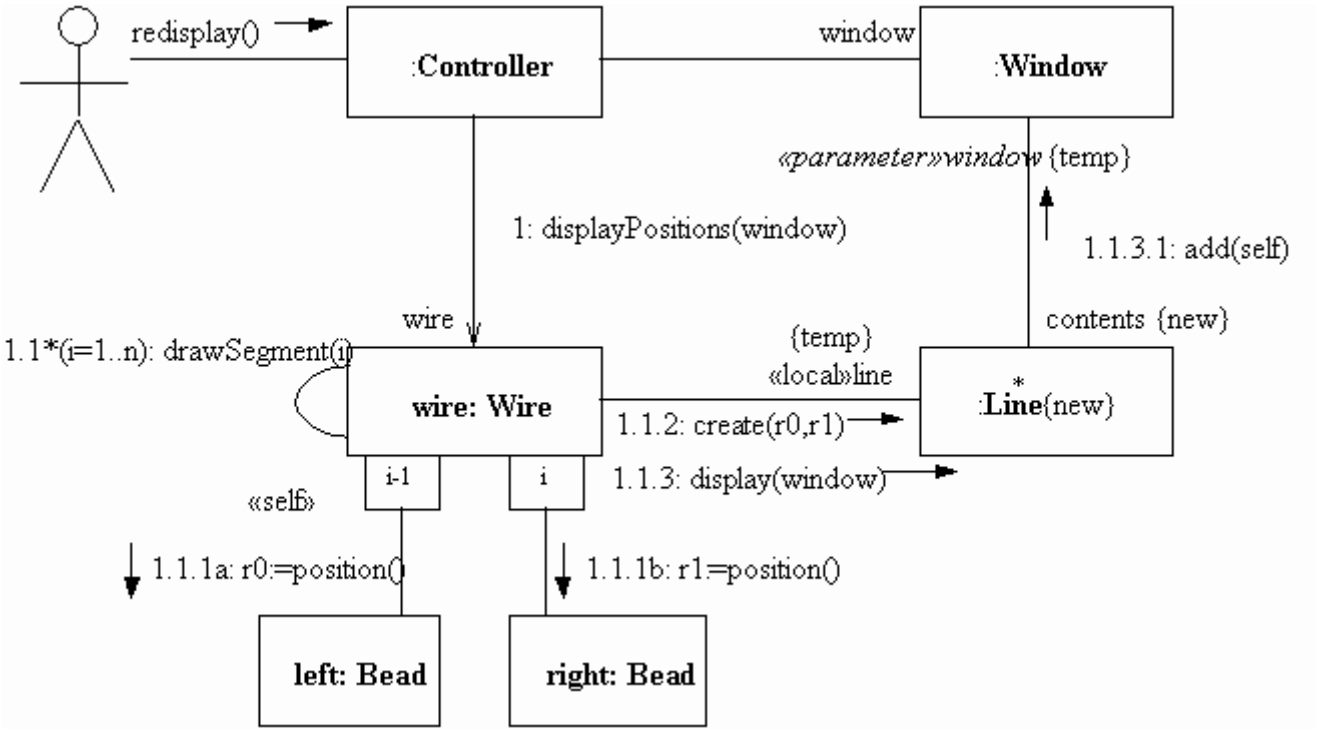


Рисунок 3 – Кооперативная диаграмма

3. Выполнение работы

1. Изучить раздел 1 и 2.
2. Изучить раздел 3. Выполнить упражнения 1 – 4.
3. Разработать модель поведения системы в виде диаграмм активности и последовательности в соответствии с вариантом задания.
4. Создать разработанную в п. 3 модель в среде Rational Rose.

4. Содержание отчета

В качестве отчета о выполненной работе предъявите преподавателю:

1. На экране в среде Rational Rose: упражнения 1 – 2 (раздел 3) и модель поведения системы в виде диаграмм активности и последовательности по индивидуальному заданию;
2. отчет в печатном виде содержащий:
   - задание,
   - диаграммы виде д активности и последовательности.
   -

5. Контрольные вопросы

1. Каково назначение диаграмм активности и последовательности?
2. Для чего используется диаграмма активности на стадии анализа?

3. Для чего используется диаграмма последовательности на стадии проектирования?
4. Назовите основные компоненты диаграммы последовательности.
5. Что представляет собой сообщение?
6. Что представляет собой диаграмма активности (деятельности)?
7. Назовите основные компоненты диаграммы деятельности ?
8. Каково назначение диаграммы кооперации?

6. Библиографический список

1. Вендров А.М. Проектирование программного обеспечения экономических информационных систем: Учебник. - М.: Финансы и статистика, 2002. - 352 с.
2. Вендров А.М. Практикум по проектированию программного обеспечения экономических информационных систем: Учеб. пособие. - М.: Финансы и статистика, 2002. - 192 с.
3. CASE-технологии: Практикум / Федотова Д.Э., Семенов Ю.Д., Чижик К.Н. – М.: Горячая линия – Телеком, 2003. 160 с.