Санкт-Петербургский государственный университет

Математико-механический факультет

Кафедра системного программирования

Николаева Дарья Михайловна

Управление учебным контентом на примере UML-практикума

Выпускная работа бакалавра

Допущена к защите

Зав. кафедрой:

д.ф.-м.н., проф. А. Н. Терехов

Научный руководитель:

к.ф.-м.н., доцент Д. В. Кознов

Рецензент:

ст. преп. Ю. В. Литвинов

Санкт-Петербург

2014

Saint Petersburg State University

Mathematics and Mechanics Faculty

Software Engineering Department

Daria Nikolayeva

Learning content management: assignments for UML workshop

Bachelor's thesis

Admitted for defense.

Head of Department:

Professor A. N. Terekhov

Scientific Advisor:

Ph.D., associate professor D. V. Koznov

Reviewer:

Senior Lecturer Y. V. Litvinov

Saint Petersburg

2014

Оглавление

Введение

Одним из способов наглядного представления информации о программном обеспечении является визуальное моделирование — метод, который, согласно [1]:

- использует графовые модели для визуализации ПО;
- предлагает моделировать ПО с разных точек зрения;
- может применяться в разработке и эволюции ПО, а также в различных видах деятельности по его созданию.

Полученные в результате визуального моделирования графовые модели могут использоваться при обсуждениях основных аспектов ПО при его разработке с различными заинтересованными сторонами, а также в формальных спецификациях ПО и в документах, делая последние более понятными и выделяя главную информацию.

Наиболее известным и широко используемым языком визуального моделирования является UML (Unified Modeling Language — унифицированный язык моделирования) [2]. Он позволяет описывать требования, бизнес-процессы, структуру ПО, архитектуру и алгоритмы. Используя один стандарт, известный по всему миру, можно решать очень широкий спектр задач. Однако здесь же кроется и главный недостаток UML — обилие разнообразных средств, включённых в него, а также внушительные размеры описания (около 1000 страниц) делают язык достаточно сложным для изучения [3], [4], поэтому часто его практическое использование ограничивается небольшим подмножеством. Так в работе [5] в качестве такого подмножества указываются диаграммы вариантов использования, классов и сценариев.

UML посвящено немало книг и пособий, которые, однако же, преимущественно излагают лишь теорию и примеры. Но в языке имеется много тонкостей, которые можно освоить лишь на практике. Отсюда появляется необходимость создания учебного практикума, который помогал бы осваивать разные неочевидные аспекты UML и грамотно использовать язык.

В настоящее время активно развивается электронное обучение (e-learning), предполагающее использование информационных и коммуникационных технологий, а также мультимедийных материалов[1]. Для разработки учебных онлайн-материалов, управления ими, а также их распространения с обеспечением совместного доступа были созданы системы управления обучением — LMS (Learning Management System). С их помощью организовывается эффективное взаимодействие между преподавателями и

[1] E-learning: http://en.wikipedia.org/wiki/E-learning

студентами, что позволяет как решать задачи дистанционного обучения, так и организовывать пространство обычных занятий в классе.

Так появляется идея о том, чтобы не только создать учебный практикум по изучению UML, но и адаптировать его для использования в средствах электронного обучения.

1. Постановка задачи

Цель данной работы состоит в том, чтобы реализовать электронный практикум по UML, используя современные концепции и средства электронного обучения. Основная особенность данного электронного практикума – настраиваемость на индивидуальные особенности обучаемых (имеющиеся знания, мотивация и др.).

В рамках работы ставятся следующие задачи:

- доработка UML-практикума по исходным материалам;
- изучение электронных систем управления обучением;
- реализация онлайн-сервиса для выборки заданий из практикума, интеграция с существующей электронной системой обучения;
- апробация метода разработки и контроля дипломных работ средствами интеллект-карт (mind maps).

2. Обзор

2.1 Краткое описание UML

В данном разделе будет кратко описан язык UML, при этом используются материалы и рисунки работы [1].

Язык визуального моделирования UML представляет из себя формализованный набор графических символов и правил построения из них визуальных моделей.

Авторы UML выделяют 13 типов диаграмм – см. рис. 1. Стоит отметить, что узлы «Структурные», «Поведенческие», «Взаимодействий» обозначают не конкретный тип, но группу диаграмм.

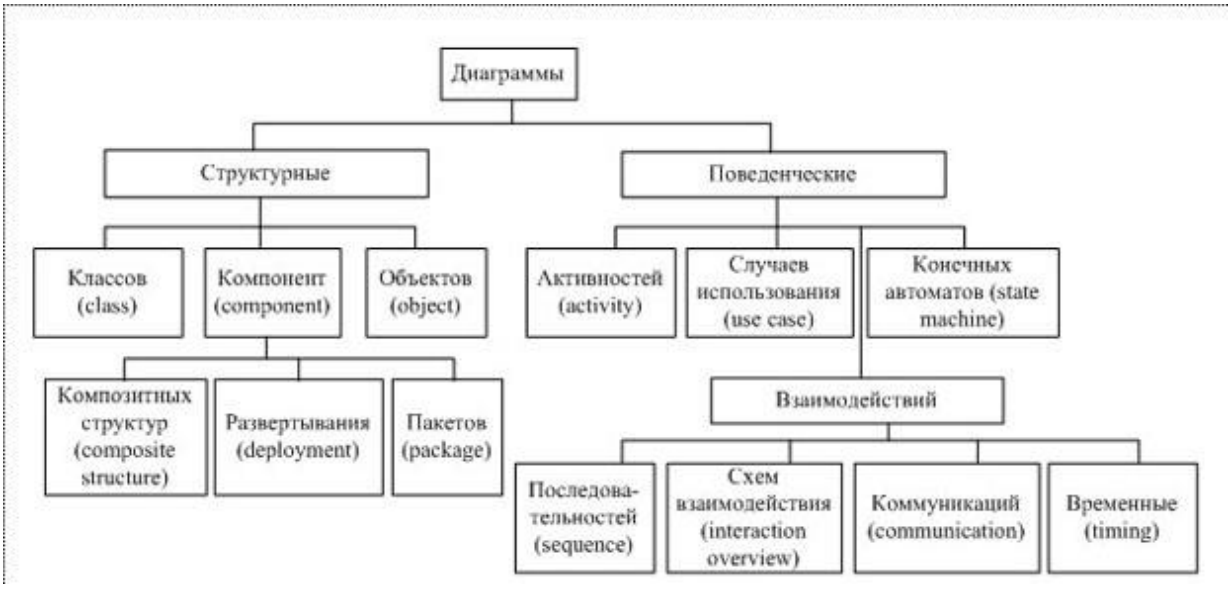

Рис. 1. Типы диаграмм UML 2.0

При разработке объектно-ориентированной системы основным типом диаграмм являются диаграммы классов, позволяющие наглядно изобразить структуру классов приложения. Такие диаграммы позволяют зафиксировать следующую информацию: название класса, его атрибуты, типы атрибутов, методы, типы и параметры метода, взаимодействие классов друг с другом (в том числе иерархию наследования) – при помощи различных видов связей.

UML поддерживает следующие типы связей между классами:

- ассоциация;
- агрегирование;
- наследование.

Наличие между классами связи-ассоциации говорит о том, объекты этих классов каким-либо образом связаны друг с другом (вызывают методы друг друга, работают с общей памятью и др.).

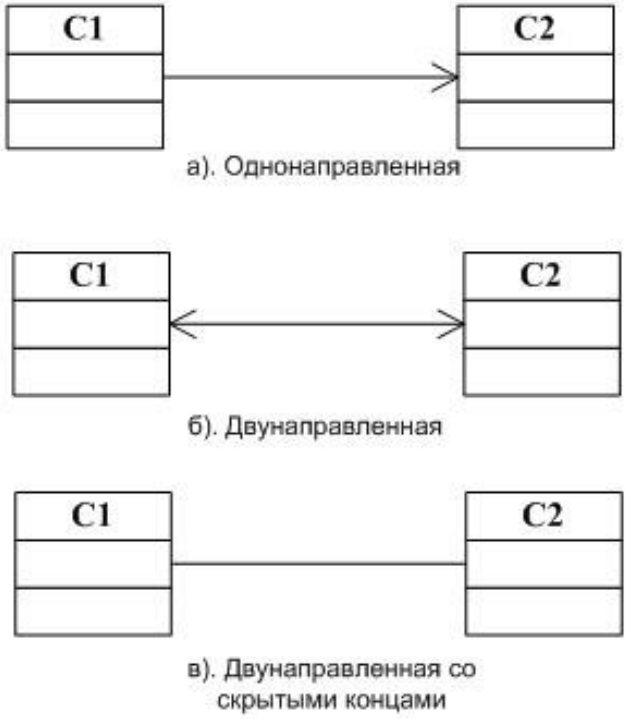

Рис. 2. Виды ассоциаций

Агрегирование – это частный случай ассоциации, который обозначает наличие связи «целое/часть» между экземплярами классов. Объект-часть в той или иной форме включается в объект-целое.

Наследование является отношением только между классами и не переходит в отношение между экземплярами классов, в отличие от ассоциации и агрегирования. Наследование модифицирует класс-предок, добавляя в него новые свойства (атрибуты, методы, реализацию методов).

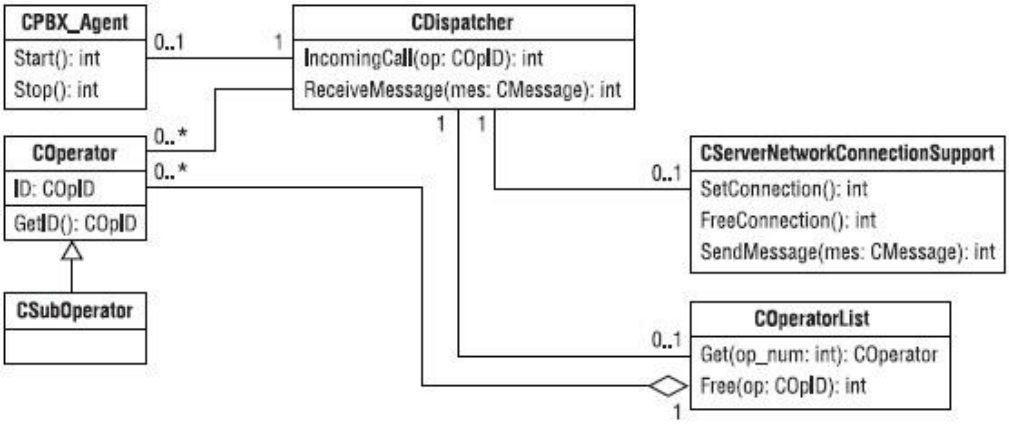

Рис. 3. Пример диаграммы классов

Для описания фрагмента системы при помощи экземпляров классов – объектов – используется одноимённая диаграмма. Объект – это конкретный экземпляр некоторого класса, обладающий уникальным идентификатором, который позволяет отличить его от других объектов этого же класса, а также конкретными значениями атрибутов и связей. Пример диаграммы объектов – на рис. 4.

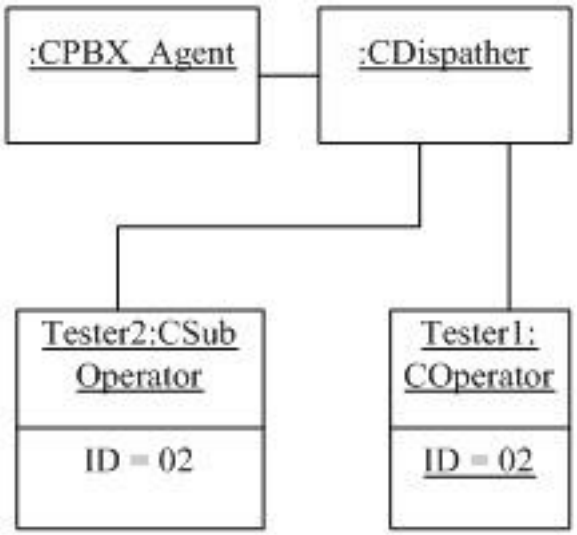

Рис. 4. Пример диаграммы объектов.

Требования к системе удобно отобразить на диаграмме случаев использования. Элементами диаграммы являются потенциальные пользователи системы, именуемые актёрами (actors), и функции системы (случаи использования), необходимые пользователю. Пример диаграммы приведён на рис. 5.

2.2 UML-практикум

В этом разделе описаны исходные материалы по UML-практикуму, которые в рамках данной работы было необходимо доработать.

Студенты кафедры системного программирования СПбГУ знакомятся с основами UML в рамках отдельного курса, посвящённого изучению этого языка моделирования. Особый акцент делается на практической части, специально для которой автором курса были составлены задачи, объединённые в практикум.

Каждое из заданий нацелено на отдельный небольшой аспект UML, нуждающийся, как это показывает многолетний опыт автора курса, в особом акценте. В практикум вошло несколько контрольных вопросов, которые носят теоретический характер, и практические задачи, для решения которых требуется нарисовать соответствующую диаграмму. Практикум охватывает диаграммы классов и диаграммы объектов.

Практикум охватывает следующие аспекты диаграмм классов:

- отношение агрегирования;
- иерархия агрегирования;
- наследование;
- правила построения иерархии наследования;
- совместное использование агрегирования и наследования;
- множественное наследование и множественность классификаций;
- однонаправленные ассоциации;
- множественность концов ассоциаций;
- множественное агрегирование;
- имена ролей ассоциаций;
- рефлексивные ассоциации (одно- и двунаправленные).

Предполагается, что студенты обладают следующими навыками и знаниями:

- основы программирования на языках Java или C#;
- минимальный опыт работы с базами данных;
- сведения из области компиляторов и формальных грамматик;
- первичные навыки написания технической документации.

Выполнение этих требований необходимо для понимания студентами предметной области предлагаемых задач.

Возможные случаи использования практикума приведены на рис. 5.

Рис. 5. Диаграмма случаев использования практикума

2.3 Электронное обучение и LMS-системы

Электронное обучение, получившее широкое распространение за рубежом, в настоящий момент активно развивается и в России. К его достоинствам можно отнести следующие:

- свобода доступа к материалам онлайн-курса: это возможно везде, где есть доступ к сети интернет;
- гибкость обучения: студент самостоятельно выбирает курсы в соответствии со своими потребностями и возможностями;
- гибкий график, возможность обучения без отрыва от работы;
- отслеживание студентом своих успехов.

Для удобства организации электронного обучения были созданы специальные платформы – системы управления обучением (в русском переводе английской аббревиатуры LMS часто фигурирует понятие «система дистанционного обучения», или СДО).

С помощью системы студент получает возможность выбрать интересующий его курс и записаться на него, настроить график обучения под свои потребности, после чего ему предоставляется доступ к материалам и заданиям курса.

Преподаватели формируют методическую базу курса, настраивают её для задач отдельного студента или группы студентов, создают задания для контроля знаний и проверяют успеваемость студентов. Как правило, и студент, и преподаватель могут отслеживать учебные успехи первого.

Административные функции систем управления обучением включают в себя задачи регистрации пользователей, контроля их доступа к системе, организации слушателей в группы для предоставления совместного доступа к учебным материалам и формирования отчётности, организации коммуникации между студентами и преподавателями.

Помимо прочего, СДО поддерживают возможность объединения традиционного (аудиторного) формата обучения с виртуальным.

Международной основой обмена электронными курсами является сборник спецификаций и стандартов, разработанный для СДО инициативной группой ADL (Advanced Distributed Learning – расширенное распределённое обучение), – SCORM (Sharable Content Object Reference Model – образцовая модель обмена учебными материалами) [6]. Он содержит требования к организации учебного материала и всей системе дистанционного обучения.

2.4 Интеллект-карты и Comapping

Содержание данного раздела излагается по источнику [7].

В 1970-х гг. английский психолог Тони Бьюзен предложил новый способ работы с информацией. Идея Бьюзена состоит в том, чтобы визуализировать данные при помощи интеллект-карт, или карт памяти (mind maps) [8]. Интеллект-карта представляет из себя диаграмму с очень простой нотацией. Диаграмма имеет древовидную структуру, в узлах которой находятся понятия, идеи, задачи, отходящие от центрального элемента – ключевой концепции, идеи – и поясняющие, детализирующие его. Основное достоинство использования карт памяти состоит в лёгкости восприятия и запоминания информации.

Один из многочисленных инструментов для работы с интеллект-картами – веб-приложение Comapping [9]. Этот продукт был разработан при участии компаний Area9, ЗАО «Ланит-Терком», а также Санкт-Петербургского государственного университета.

Для представления карт памяти Comapping использует древовидную нотацию, в которой уточнение концепций происходит слева направо (т.н. left-to-right mindmapping) и, соответственно, главный элемент карты памяти находится левее остальных. Comapping позволяет связывать с элементами интеллект-карты заметки и файлы, производить проверку правописания текста в узлах карты, осуществлять текстовый поиск и фильтрацию элементов карты, публиковать карты на сайтах, экспортировать их в форматы PDF, HTML, RTF, SVG, импортировать/экспортировать в форматы других средств работы с картами памяти (FreeMind, MindManager).

Кроме того, comapping предоставляет функционал для совместной работы пользователей: возможность одновременного просмотра и редактирования карты, чат для внутреннего общения, настройки различных прав доступа, сохранение истории изменений с указанием пользователей, которые их вносили, а также даты и времени.

2.5 Метод проектирования дипломных работ с использованием карт памяти

С 2011 года на кафедре системного программирования СПбГУ проходит обязательный для всех выпускников курс «Практика разработки документации», направленный на то, чтобы помочь студентам грамотно зафиксировать результаты своей выпускной работы. Поскольку тексты дипломных работ являются одним из главных критериев оценки эффективности работы университетских кафедр, важно уделить внимание качеству этих текстов.

В рамках данного курса применяется новая методика проектирования дипломной записки с использованием карт памяти и средства для их создания – Comapping. Курс состоит из групповой и индивидуальной частей (рис. 6). В ходе групповой части

студентов знакомят с возможными вариантами структуры текста выпускной работы, объясняют, на чём сделать акцент в зависимости от характера работы, а также рассказывают о необходимых для работы функциях Comapping. После этого студентам предлагается при помощи карт памяти составить подробный план своей дипломной записки. Здесь начинается индивидуальная работа с каждым студентом: преподаватель, который ведёт курс, отдельно оценивает каждую карту памяти, предлагает варианты её улучшения. Важно отметить, что в узлах интеллект-карт могут и должны быть не условные обозначения, а полноценные фразы, предложения, которые потом станут заголовками глав или частями текста.

В процессе создания карты памяти рекомендуется участие научного руководителя для внесения корректив и уточнения деталей, на которых в выпускной работе необходимо сделать акцент. Таким образом, к концу курса студент имеет подробный план, на основании которого может писать выпускную работу.

Для оценки эффективности применяемого в рамках курса метода был проведён анализ дипломных работ выпускников кафедры системного программирования СПбГУ, описанный в главе 5.

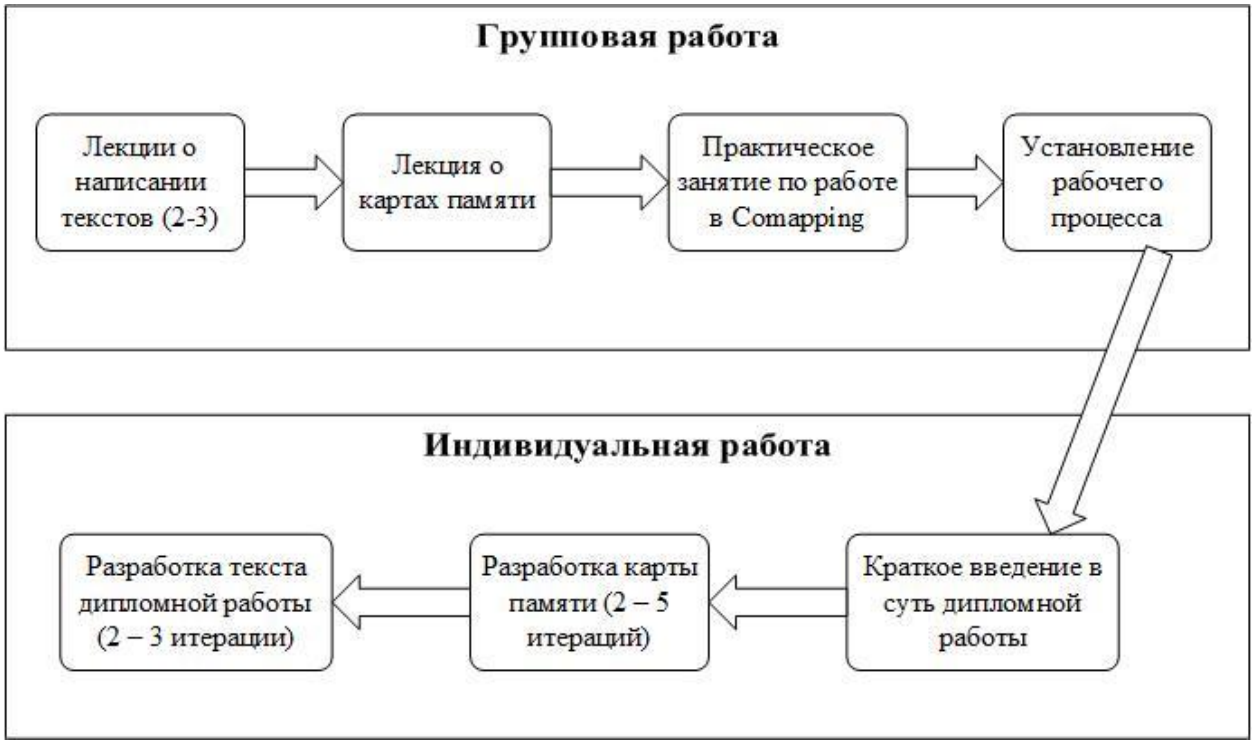

Рис. 6. Схема курса «Практика разработки документации»

3. Доработка практикума по UML

Для практических занятий студентов в рамках курса по UML был создан практикум, охватывающий диаграммы классов и диаграммы объектов. Требовалось отредактировать уже имеющиеся задания и расширить базу, дополнив её новыми заданиями на построение диаграмм классов, а также заданиями на построение диаграмм случаев использования, ранее не включённых в практикум.

При создании новых задач учитывались следующие требования.

- Каждое задание должно быть направлено на решение определённой методической задачи, при этом необходимо предельно чётко формулировать условия задания, чтобы студенты понимали, что от них ожидается.
- Каждое задание сопровождается набором тегов (рис. 7).
- Необходимо создавать несколько вариантов одного задания, которые, фактически, являются разными задачами, направленными на изучение и понимание определённого аспекта языка UML.
- Предметная область задания должна быть известна студентам: либо они прослушали соответствующий курс, либо область общеизвестна и относительно проста.

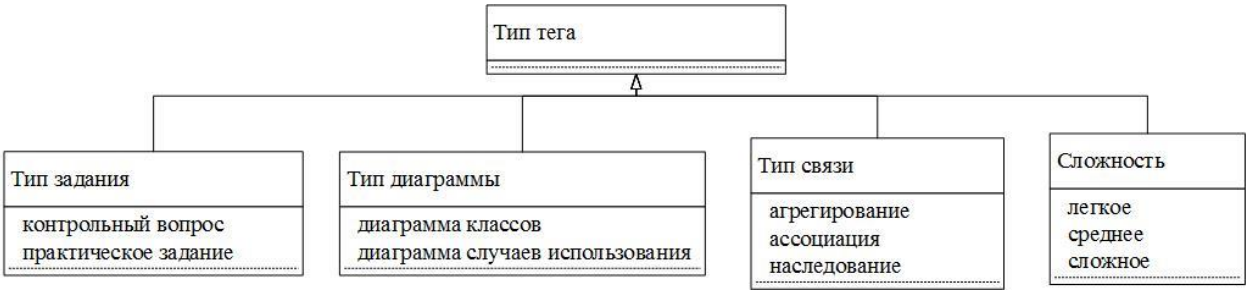

Рис. 7. Типы тегов, которыми отмечаются задания

3.1. Задания на изучение диаграмм классов

Основные трудности при построении диаграммы классов – это выбор того, какие детали предметной области должны стать отдельными классами, какие – атрибутами класса, определение правильных связей между классами, а также чёткое понимание отличий между ассоциацией, агрегированием и наследованием.

Тематика заданий по диаграммам классов, вошедших в практикум, упоминалась в разделе 2.2. В рамках данной дипломной работы тематика не расширялась, только добавлялись новые задачи по уже имеющимся темам. Ниже рассмотрим их более подробно.

Для задач использовались преимущественно простые предметные области, поскольку разобраться с новым материалом на понятном примере легче.

Задачи на агрегирование (наследование) отмечены тегом «лёгкие», с них рекомендуется начинать знакомство студентов с UML и построением диаграмм классов.

Пример задач на агрегирование.

Требуется выделить необходимые сущности и связи между ними и нарисовать диаграммы классов для следующих ситуаций:

1. *Банк состоит из различных филиалов, а также головного офиса.*
2. *Университет состоит из разных факультетов.*

На рис. 8, 9 представлены решения для этих задач.

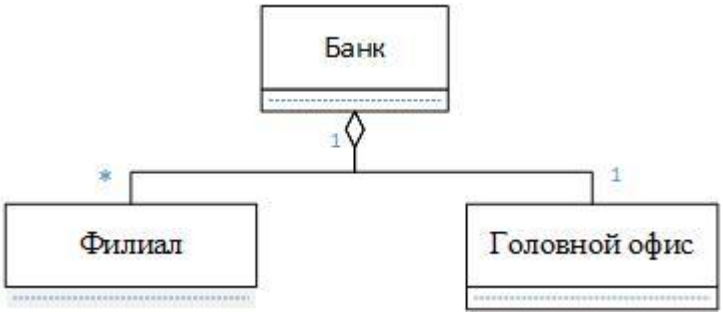

Рис. 8. Решение задачи на агрегирование №1

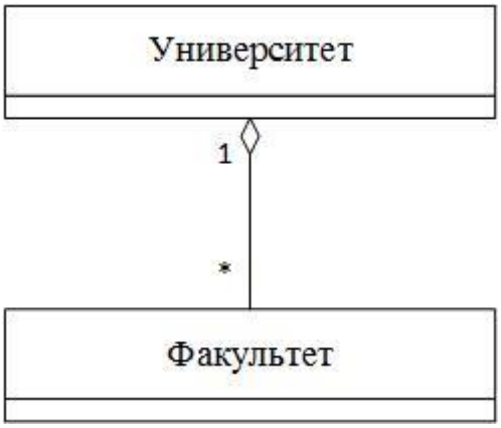

Рис. 9. Решение задачи на агрегирование №2

Задачи на иерархию агрегирования (наследования) являются усложнением предыдущих задач (на агрегирование и наследование соответственно).

Подавляющее большинство тех, кто начинает работать с UML, сталкивается с проблемой понимания разницы между наследованием и агрегированием. Эти два вида связи действительно имеют много общего: оба позволяют строить древообразную иерархию классов, изображения обоих отношений чем-то похожи визуально, предок, как

и агрегируемый класс, добавляет функциональность в потомок/агрегат. Однако это всё же два разных отношения, отличия между которыми упоминались в разделе 2.1. Для того, чтобы студенты смогли лучше это понять, в практикум включены задачи на совместное использование агрегирования и наследования, наглядно демонстрирующие разницу между ними.

Пример задач на совместное использование агрегирования и наследования.

Выделите необходимые сущности и связи между ними и нарисуйте диаграмму классов для следующих ситуаций. Пользуйтесь агрегированием и наследованием, не пользуйтесь атрибутами.

1. *Есть лес, в нем растут деревья – сосны, березы, ивы. Березы бывают следующих видов: береза бумажная, береза вишневая, береза даурская. Сосны бывают следующих видов сосна чешуйчатая, сосна уэмацу, сосна юньнаньская. У каждого дерева есть ствол, ветви, коневая система. Еще в лесу живут птицы – синицы, дрозды, совы (ушастая и болотная).*
2. *Банк состоит из различных филиалов, а также головного офиса. Все подразделения банка состоят из департаментов. Департаменты бывают производственными и административными. В департаментах работают сотрудники.*

На рис. 10, 11 представлены решения для задач из этого примера.

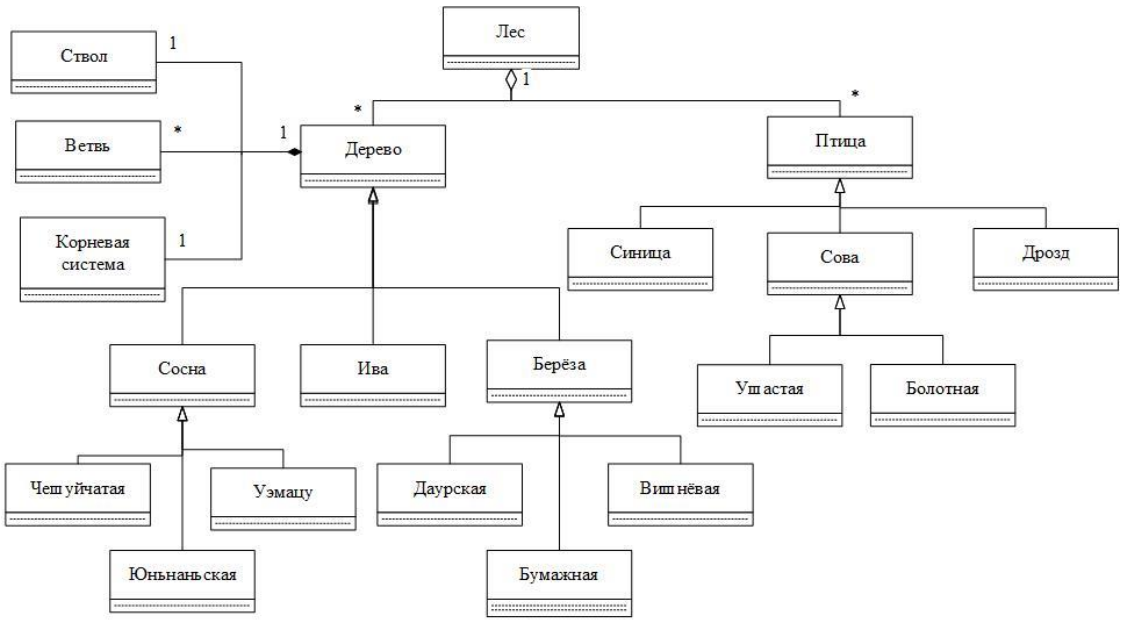

Рис. 10. Решение задачи на совместное использование наследования и агрегирования №1

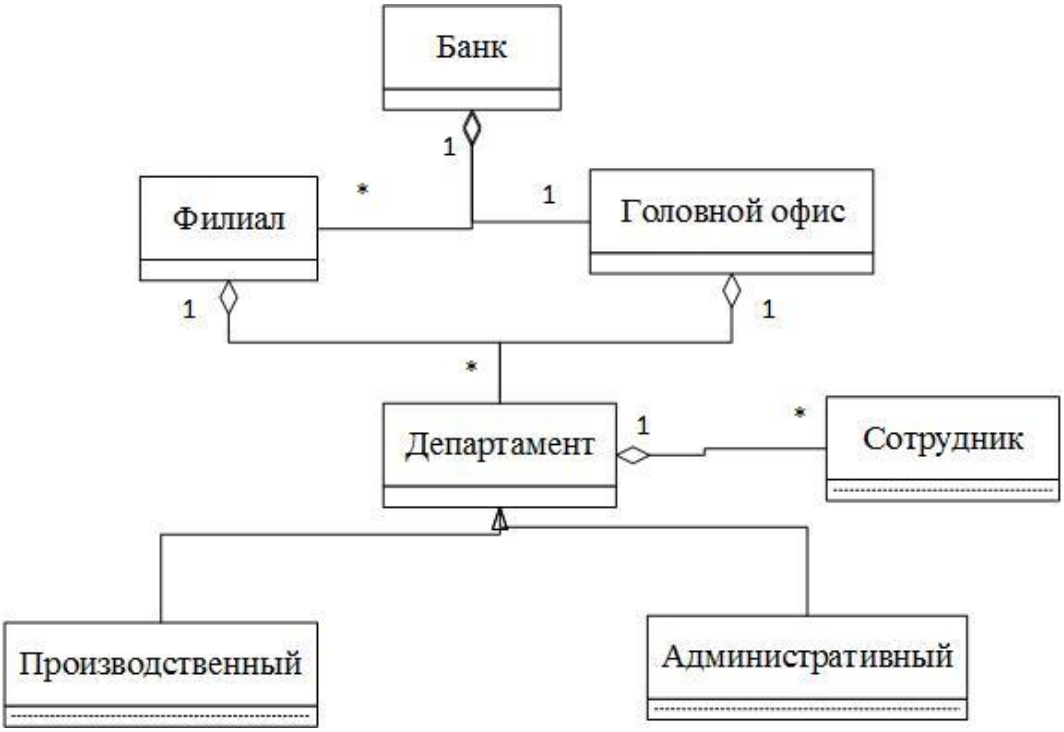

Рис. 11. Решение задачи на совместное использование наследования и агрегирования №2

Задачи на наследование и множественность классификаций являются усложнённой модификацией задач на иерархию.

Задания на однонаправленную связь, множественность концов ассоциаций, множественное агрегирование, имена ролей и рефлексивную ассоциацию сформулированы таким образом, чтобы студент непременно использовал требующееся в задании отношение или свойство этого отношения, формируя представление о том, в каких случаях необходимо применять именно их. Кроме того, эти задания заставляют студента рассуждать, поскольку являются более сложными, чем представленные ранее.

Пример задачи на имена ролей.

Нарисуйте диаграммы классов для следующей ситуации, используя имена ролей. Будьте внимательны также к значениям множественности концов ассоциаций.

1. *В проекте обязательно есть менеджер, один или несколько разработчиков, один или несколько тестировщиков, один или несколько технических писателей, маркетолог. При этом менеджер и разработчики в один момент времени могут участвовать только в одном проекте, а тестировщики, технические писатели и маркетолог — в нескольких (но не меньше, чем в одном). Используйте только классы «Проект» и «Сотрудник».*

На рис. 12 представлено решение для задачи из этого примера.

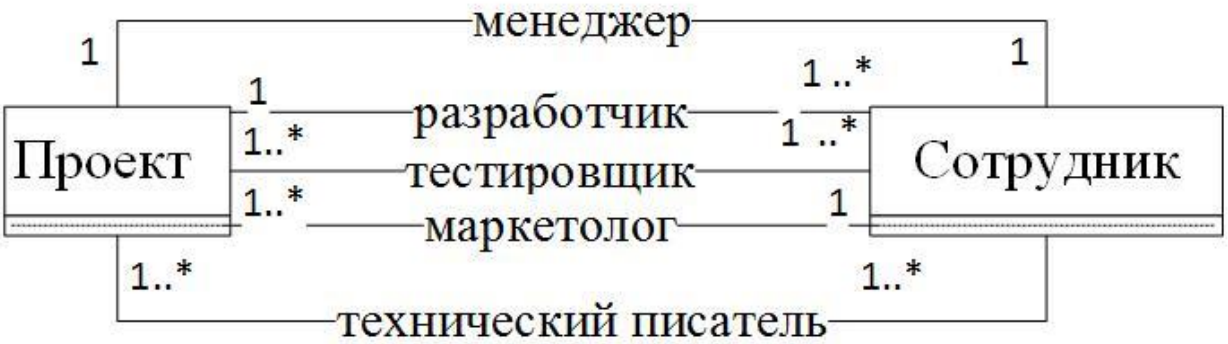

Рис. 12. Решение задачи на имена ролей №1

3.2. Задания на изучение диаграмм случаев использования

Диаграммы случаев использования значительно более просты для понимания и освоения, чем диаграммы классов ввиду отсутствия такого большого количества языковых конструкций. Главная сложность при их использовании заключается в том, чтобы научиться грамотно выделять потенциальных пользователей – актёров – и формулировать их требования (ожидания) к системе.

В связи с этим задания составлялись следующим образом: для первого знакомства с диаграммами случаев использования выбирается простая предметная область, для которой выделяется 2-3 актёра (для облегчения задачи они уже указаны в тексте задания). Студенту предлагается сформулировать для каждого из актёров минимальный набор требований и отобразить их на диаграмме. Повышение сложности заданий характеризуется выделением бо́льшего количества пользователей и, соответственно, бо́льшего количества требований, усложнением предметной области, а также предложением выделить актёров самостоятельно (в предыдущих задачах актёры задавались прямо в условии задания).

Пример простого задания.

Нарисуйте диаграмму случаев использования для следующей системы, используя указанных актёров. Самостоятельно сформулируйте для каждого актёра минимальный набор требований к системе.

1. ***Система «Электронная библиотека».*** *Зарегистрированный пользователь – читатель – получает доступ к материалам библиотеки (просмотр, скачивание отдельных, отмеченных специальным знаком, статей). Актёры: администратор системы, читатель (зарегистрированный пользователь), незарегистрированный пользователь.*

На рис. 13 представлено решение для задачи из этого примера.

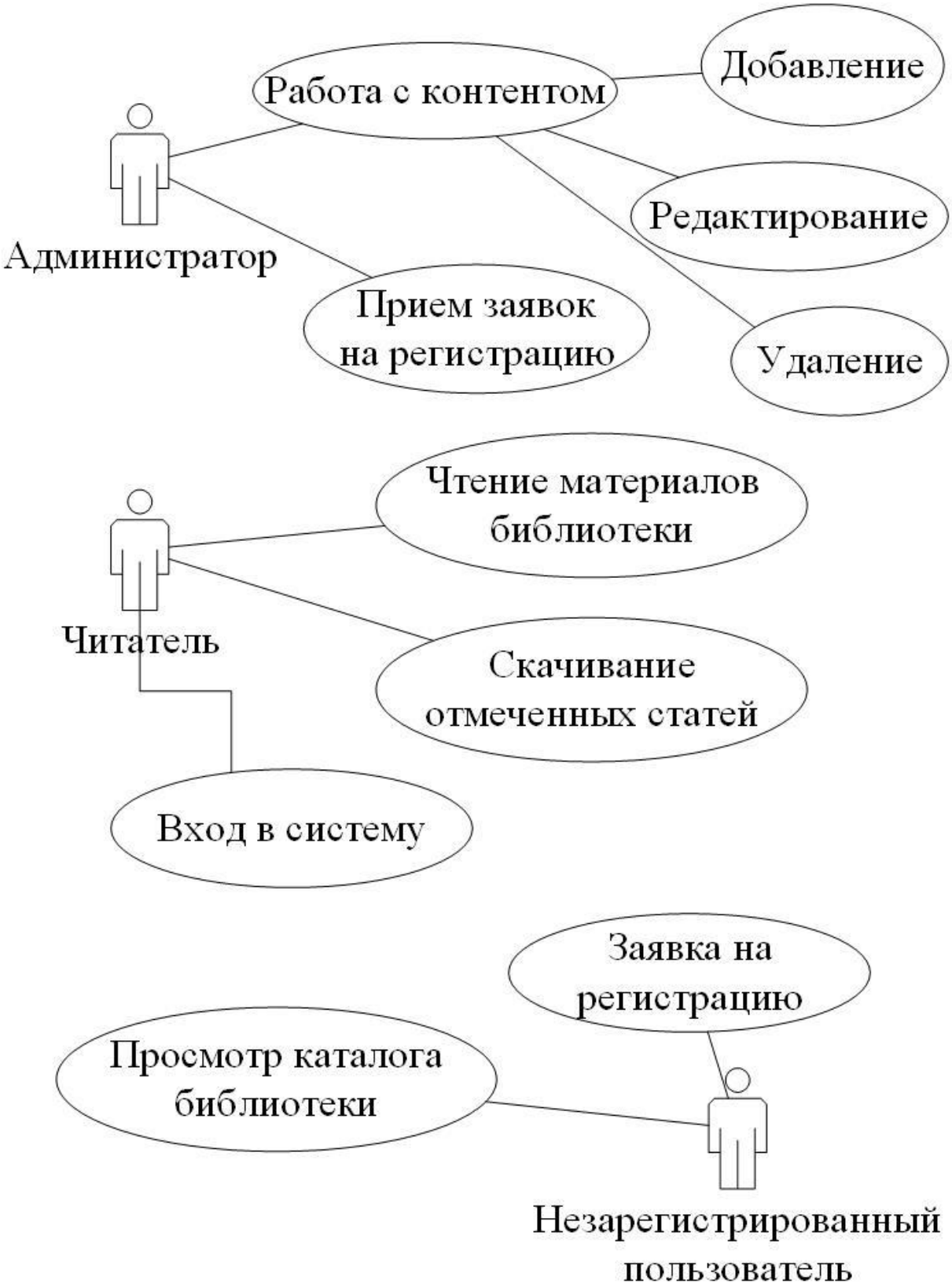

Рис. 13. Решение простой задачи на построение диаграммы случаев использования

Пример сложного задания.

Нарисуйте диаграммы случаев использования для следующих систем. Самостоятельно выделите пользователей для каждой системы, сформулируйте для них требования к системе.

1. ***Организация концерта музыкальной группы.*** *Необходимо найти концертную площадку, отвечающую техническому райдеру музыкантов, привезти группу и предоставить им проживание в соответствии с бытовым райдером, организовать продажу билетов, рекламу.*

На рис. 14 представлено решение для задачи из этого примера.

Рис. 14. Решение сложной задачи на построение диаграммы случаев использования

4. Электронный сервис по настройке практикума

Описанный в предыдущем разделе UML-практикум представляет из себя набор заданий различной сложности и тематики, из которого преподаватель производит выборку задач для конкретной учебной группы с учётом особенностей студентов (прослушанных курсов, мотивации, интеллектуальных способностей и т.д.) для того, чтобы сделать обучение наиболее эффективным. Главная цель разрабатываемого сервиса – автоматизировать процесс создания такой выборки, которая в дальнейшем интегрируется с одной из используемых в СПбГУ систем дистанционного обучения – Sakai.

4.1. Требования

Система должна предоставлять следующие функциональные возможности:

- добавление новых заданий в базу;
- редактирование добавленных ранее заданий;
- удаление заданий из базы;
- осуществление поиска заданий по тегам;
- экспорт заданий в файл в формате txt.

Добавление новых заданий должно осуществляться путём заполнения формы, содержащей следующие поля:

- название задания;
- описание задания (формулировка того, что нужно сделать);
- текст задания;
- теги.

Основные требования к системе представлены на рис. 15.

4.2. Реализация

Для реализации системы была выбрана технология ASP.NET с применением фреймворка ASP.NET MVC 4, который используется для разработки веб-приложений на основе шаблона MVC (Model – View – Controller). В качестве СУБД использовалась Microsoft SQL Server. Для работы с объектами базы данных применялся Entity Framework – ORM решение (Object-Relational Mapping – объектно-реляционное отображение) для платформы .NET от Microsoft.

Рис. 15. Требования к сервису по настройке практикума

Для создания базы данных применялась методика разработки Code First, которая предполагает, что сначала описываются классы-модели, после чего фреймворк (в данном случае – Entity Framework) автоматически генерирует по ним базу данных.

Описываемый сервис содержит модель UmlTask, которая обладает следующими свойствами:

- int Id – уникальный идентификатор задания, генерируется автоматически;
- string Title – название задания;
- string TaskDescr – описание задания;
- string TaskContent – текст задания;
- string TagTaskType – тег «Тип задания»;
- string TagDiagramType – тег «Тип диаграммы»;
- string TagRelationshipType – тег «Тип связи»;
- string TagComplexity – тег типа «Сложность».

Для подключения к базе данных создан класс UmlTaskDataContext, производный от класса DbContext. Он содержит свойство UmlTasks типа DbSet<UmlTask>, с помощью которого можно получить доступ к данным соответствующей модели (UmlTask).

Класс-контроллер UmlTaskController содержит методы, позволяющие осуществлять работу с контентом (добавление, редактирование и удаление данных) и поиск по базе.

На рис. 16 показана главная страница сервиса, на которую выводятся основные ссылки и задания (название, описание и теги). Просмотреть задание целиком можно, перейдя по ссылке в названии.

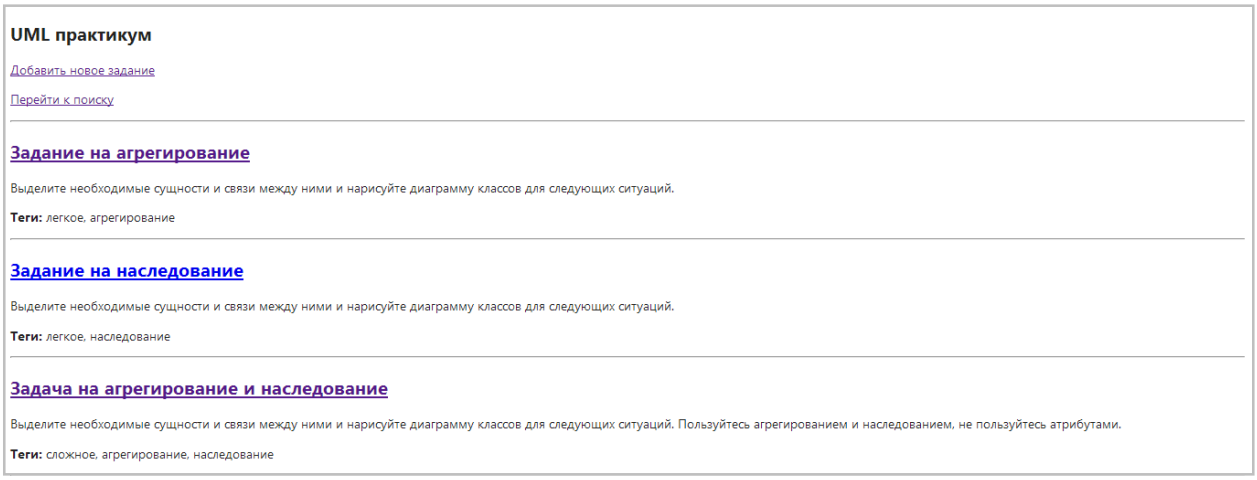

Рис. 16. Главная страница сервиса

На рис. 17 представлена страница добавления нового задания в базу.

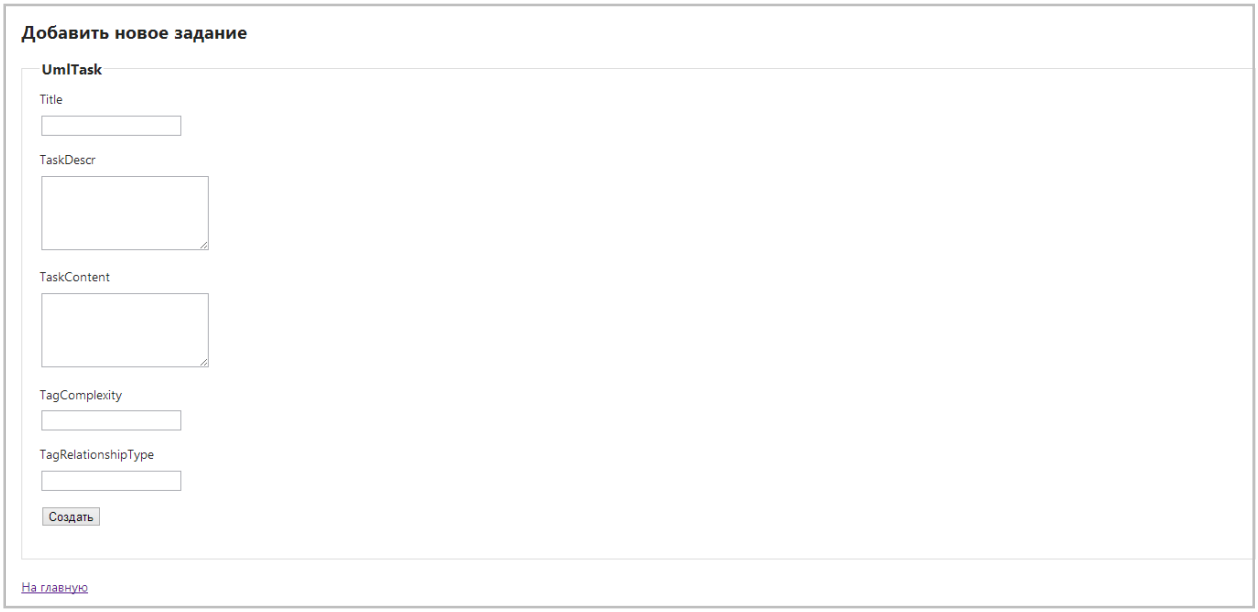

Рис. 17. Страница добавления нового задания в базу данных

Для поиска необходимо отметить нужный тип тега и ввести название тега в поисковую строку – рис. 18.

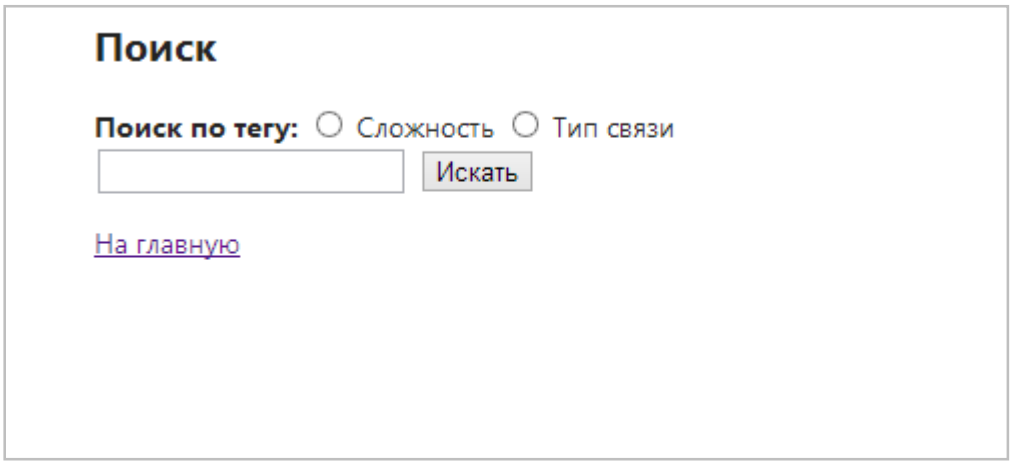

Рис. 18. Страница поиска по тегам

5. Исследование эффективности использования карт памяти при разработке дипломов

Как уже упоминалось ранее, на кафедре системного программирования проходит курс, в рамках которого студенты составляют подробные планы текстов своих дипломных работ, используя карты памяти. Как следует из рис. 19, в среднем кафедра выпускает по сорок студентов в год. Для того, чтобы отслеживать качество дипломных работ такого количества студентов, требуются специальные методы.

Для оценки эффективности курса был проведён анализ дипломов, написанных выпускниками в период с 2007 по 2009 год, когда курс ещё не проводился, и в период с 2011 по 2013 годы, после его введения. Рассматривались работы специалистов, поскольку бакалавры и магистры ещё не составляют стабильного потока, и, кроме того, они пишут уже второй диплом, в силу чего им значительно легче.

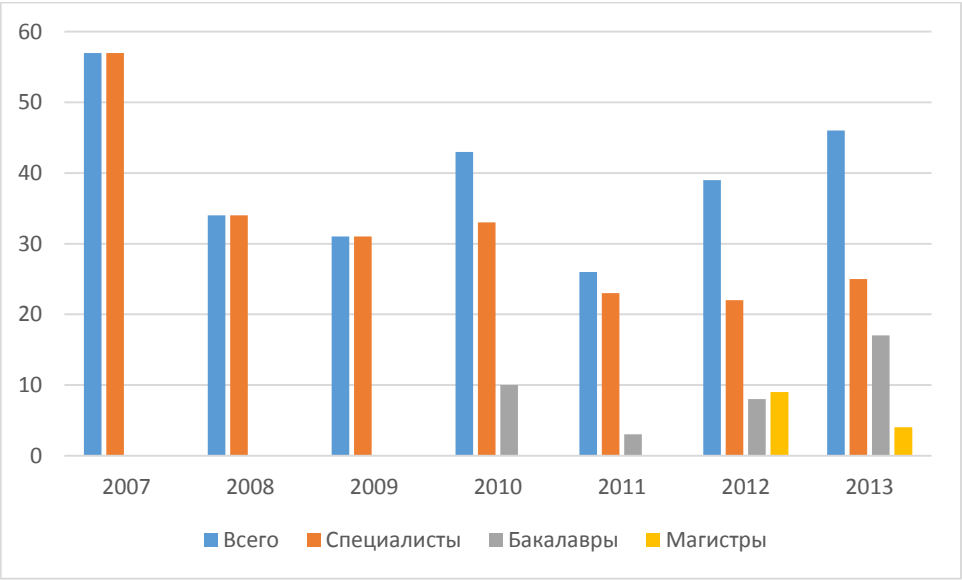

Рис.19. Статистика количества выпускников кафедры системного программирования СПбГУ за 2007 – 2013 гг.

Анализ текстов дипломов проводился по ряду критериев, представленных ниже. После каждого критерия приведён график, показывающий изменение средней оценки с течением времени.

Критерии оценивают главным образом структуру текста, не касаясь его качества в целом. Все данные по объёму работ были приведены к единому рекомендуемому формату (шрифт Times New Roman 14, межстрочный интервал 1,5).

5.1 Критерии оценки дипломных работ

В данном разделе описаны критерии, использованные для оценки качества текстов дипломных работ

1. **Качество введения** (соответствующая статистика по всем дипломам из исследуемой выборки представлена на рис. 20).
    - 5 – в дипломе имеется подробное введение в предметную область, постепенно раскрывающее различные контексты работы, аргументирующее актуальность/новизну тематики и подводящее читателя к постановке задачи.
    - 4 – введение слишком маленькое (меньше одной страницы), не в полной мере присутствует один из параметров, указанных выше, или если во введении присутствуют элементы литобзора.
    - 3 – введение не содержит нет связного и постепенного вступления в предметную область работы, или не содержит обоснования актуальности задачи, или если объём введения значительно превосходит норму (в качестве нормы мы считали объём введения, равный трём страницам).
    - 2 – введение подменяется литобзором или подробной постановкой задачи.
    - 1 – введения нет.

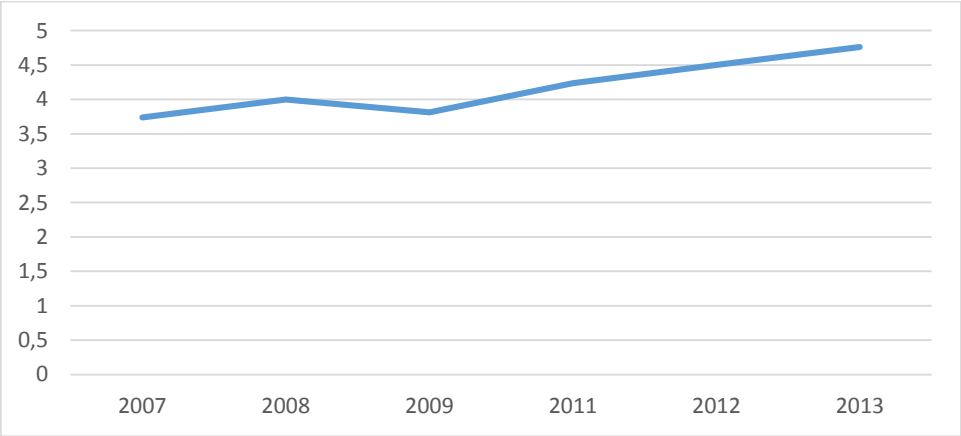

Рис. 20. Изменение средней оценки качества введения

2. **Качество постановки задачи и результатов (заключение)** – соответствующая статистика по всем дипломам из исследуемой выборки представлена на рис. 21.
    - 5 – есть цель и раскрывающие её задачи (по пунктам), и задачам работы соответствует структура оглавления диплома.
    - 4 – есть разбиение цели на задачи, но они недостаточно конкретны, или результаты недостаточно точно соответствуют задачам, или если постановка

задачи содержится не в конце введения и не в отдельном пункте, а в каком-либо ином разделе диплома.

- 3 – все есть, но нет корреляции задач результатам и соответствия задач оглавлению, или задачи и результаты не структурированы.
- 2 – результаты не структурированы и не соответствуют поставленным задачам; цель только обозначена, но строго выделенных задач нет, нет также и ясных формулировок результатов.
- 1 – синтаксически не идентифицируются ни задачи, ни результаты, читателю нужно самостоятельно выделять все это из текста.

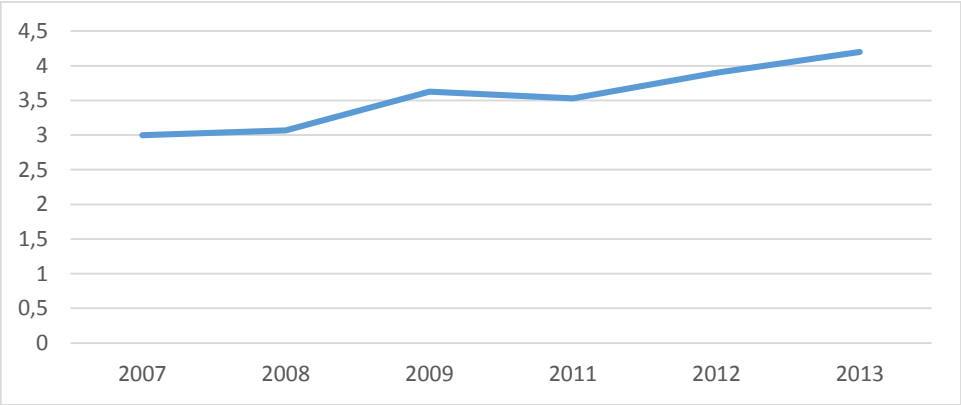

Рис. 21. Изменение средней оценки качества постановки задачи и результатов

3. **Качество литобзора** (соответствующая статистика по всем дипломам из исследуемой выборки представлена на рис. 22).
    - 5 – в литобзоре находятся все сторонние результаты, он представлен в виде отдельной главы или набора глав.
    - 4 – литобзор присутствует, но в основном тексте содержатся сторонние результаты.
    - 3 – сторонние результаты в большом количестве встречаются вне литобзора, или последний недостаточно полный.
    - 2 – литобзор присутствует, но совсем небольшой и сильно неполный, или сторонние результаты и используемые технологии только упоминаются, но не описываются должным образом.
    - 1 – литобзор отсутствует в виде самостоятельной части, то есть все сторонние результаты, используемые в работе, либо вообще не описываются, либо обсуждаются в основной части.

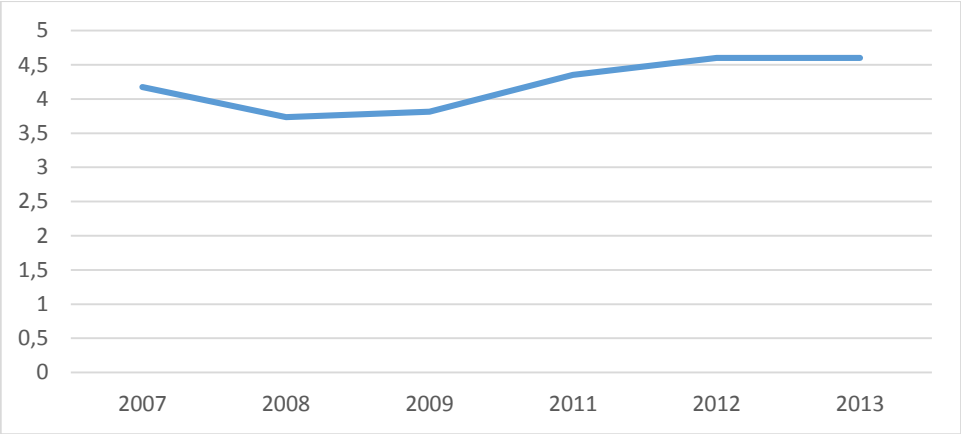

Рис. 22. Изменение средней оценки качества литобзора

4. **Сбалансированность структуры** – соответствующая статистика по всем дипломам из исследуемой выборки представлена на рис. 23. В ходе анализа дипломных работ были экспериментально выявлены оптимальные соотношения частей диплома, что представлено ниже. При этом приложения рассматривались отдельно.
    - литературный обзор должен занимать от 15% до 35% текста дипломной записки;
    - основная часть должна составлять от 40% до 50% всего текста дипломной записки;
    - служебная часть (титульный лист, оглавление, введение, постановка задачи, результаты работы и заключение, список литературы) должна составлять от 15% до 25% всего текста дипломной записки;
    - приложения по объёму не должны превосходить текста дипломной записки.

За несоблюдение каждого из указанных выше ограничений из оценки вычитается один балл (исходно она равна пяти).

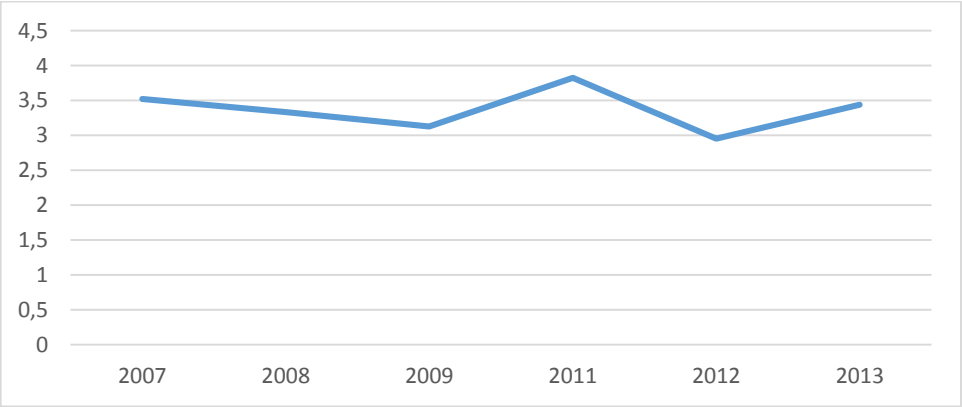

Рис. 23. Изменение средней оценки сбалансированности структуры дипломов

5.2 Анализ данных и выводы

В целом можно заметить, что оценки преимущественно растут. В дипломных работах за 2011 – 2013 годы (особенно за 2012 и 2013 гг.) ясно прослеживается единообразная структура: есть введение, постановка задачи, литобзор, основная часть и заключение с результатами работы. Почти всегда эти составные части являются отдельными главами диплома, порядок их следования соответствует указанному. В дипломах более ранних лет та или иная глава может отсутствовать вовсе либо быть представлена не в полном объёме. На рис. 24 можно увидеть изменение общей оценки качества, которая рассчитывалась как среднее между всеми за оценками за каждый год. Из этого графика видно, что введение курса – 2011 год – совпало с улучшением качества дипломных записок. Средняя оценка качества до введения курса составляла 3,65 по пятибалльной шкале, после введения – 4,1.

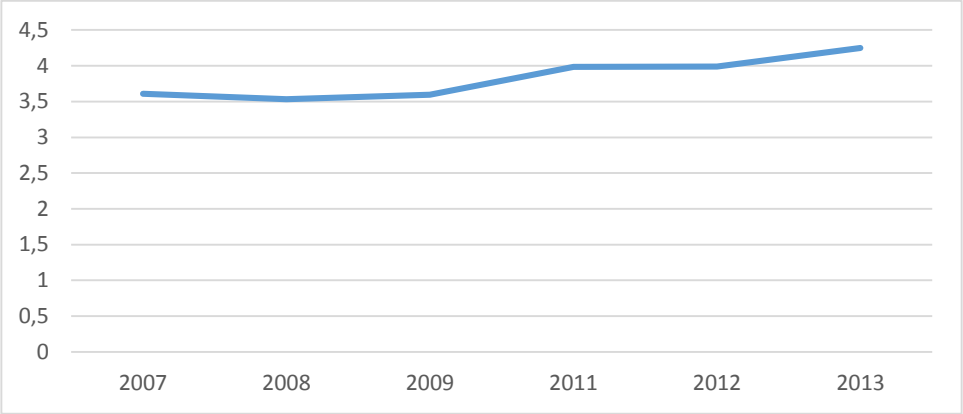

Рис. 24. Изменение общей оценки качества дипломов

В дипломах выпускников 2007 – 2009 гг. чаще происходит подмена введения постановкой задачи или обзором литературы, объём введения колеблется от ¼ страницы до 8 страниц. В 2013 году средний объём введения составил 2,5 страницы. За 2011 – 2013 гг. случаи, когда во введение был включён материал, относящийся к литобзору, встретились в 9,6% работ, за 2007 – 2009 гг. – в 22,2% работ.

Очень часто встречается ситуация, когда постановка задачи и сформулированные результаты работы слабо коррелируют друг с другом. Это общая проблема всех дипломов, хотя в 2013 году она встречается реже. Как правило, всё дело в недостаточно конкретной постановке задачи либо же в оформлении результатов, которые сами по себе могут удовлетворять поставленной задаче в полной мере.

Изменение оценки сбалансированности структуры дипломных работ говорит о том, что этому параметру не уделяется должное внимание при написании текста. Наиболее частый случай нарушения баланса – слишком большая относительно других разделов служебная часть.

Рис. 25. Гистограмма средних оценок качества за каждый год

6. Заключение

В ходе данной работы были достигнуты следующие результаты.

1. Доработан практикум UML для изучения построения диаграмм классов (23 задачи) и диаграмм случаев использования (20 задач).
2. Изучены популярные LMS Sakai, Moodle.
3. Реализован онлайн-сервис для выборки заданий из практикума (ASP.NET MVC 4, MS SQL).
4. Произведена апробация метода разработки и контроля дипломных работ средствами интеллект-карт, показавший, что в результате применения метода качество текстов дипломных работ повысилось с 3,65 до 4,1 по пятибалльной шкале.

В качестве продолжения работы предполагается расширение UML-практикума добавлением заданий для освоения других возможных видов диаграмм.

Список литературы

[1] Д. В. Кознов. Языки визуального моделирования. Проектирование и визуализация программного обеспечения. Изд-во Санкт-Петербургского университета. 2004. 150 с.

[2] OMG Unified Modeling Language (OMG UML), Version 2.5, 2013. 786 p.

[3] G. Reggio, M. Leotta, F. Ricca, D. Clerissi. What are the used UML diagrams? A Preliminary Survey. EESSMOD@MoDELS 2013. P. 3–12.

[4] B. Dobing and J. Parsons. How UML is used. Communications of the ACM, 49(5). 2006. P. 109–113.

[5] M. Grossman, J. E. Aronson, and R. V. McCarthy. Does UML make the grade? Insights from the software development community. Information and Software Technology, 47(6). 2005. P. 383–397.

[6] Advanced Distributed Learning: SCORM Version 1.2: http://www.adlnet.gov/scorm/scorm-version-1-2/

[7] Д.Кознов, Е.Ларчик, М.Плискин, Н.Артамонов. О задаче слияния карт памяти (Mind Maps) при коллективной разработке // Программирование, 2011, № 6. С. 1-10.

[8] T. Busen. The Mind Map Book. Penguin Books, 1996. 320 p.

[9] Comapping: http://www.comapping.com/