Некоммерческое акционерное общество

АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

Кафедра компьютерных технологий

УТВЕРЖДАЮ
Декан ФАИТ
_____ Табултаев С.С.
«_____»_____ 2017 г.

TRPOSRV 5301- ТЕХНОЛОГИИ РАЗРАБОТКИ ПРОГРАММНОГО ОБСПЕЧЕНИЯ ДЛЯ СИСТЕМ РЕАЛЬНОГО ВРЕМЕНИ

Методические указания к выполнению лабораторных работ
для магистрантов специальности 6М070400 - «Вычислительная техника и программное обеспечение»
(научное и педагогическое направление)

Алматы 2017

Составители: А. А. Аманбаев. Технологии разработки программного обеспечения для систем реального времени. Методические указания к выполнению лабораторных работ для магистрантов специальности 6М070400 - «Вычислительная техника и программное обеспечение» (научное и педагогическое направление). – Алматы: АУЭС, 2017. – ___ с.

Протокол № _6_ заседания кафедры от «_ 6 _» _ 06 _ 2017 г.

Заведующий кафедрой ММиПО _____ М.Ж.Байсалова

Методические указания рассмотрен и утвержден на заседании учебно-методической комиссии факультета Аэрокосмических и информационных технологий протокол № _5_ от «_20.06_» 2017 г.

Методические указания содержат указания по подготовке к проведению семи лабораторных работ основные понятия, используемые при проектировании программных систем: методы, средства, организация, технология. Содержание процесса и системы проектирования. Понятие предметной области, информационной среды. Комплекс вопросов, связанных с объектно-ориентированным подходом к проектированию программных систем, оговорен перечень рекомендуемой литературы и контрольные вопросы в конце каждой темы.

Выполнение лабораторных работ определяет изучение различных форм моделей обмена сообщениями и ее реализуемости, в том числе архитектуры с общей памятью. Рассмотрение вопросов семантики, однозначности результата и блокировки вычислений. Изучение методов анализа и ознакомление магистрантов с вопросами проектирования сложных программных систем, обучение их методологии структурного анализа и проектирования SADT, освоение ими основ объектно- ориентированного подхода к проектированию программных систем и приобретение практических навыков применения современных технологии проектирования (CASE-технологии)..

Ил. 15, библиогр. – 10 назв.

Рецензент: д-р. физ-мат. наук, проф. З.К. Куралбаев.

Печатается по плану издания некоммерческого акционерного общества “Алматинский Университет энергетики и связи” на 2017 г.

Содержание

На лабораторных занятиях и в процессе самостоятельной работы студенты должны освоить основные аспекты разработки программного обеспечения, связанные с построением конкретных моделей программных систем, и средства визуального проектирования программных моделей.

Процесс согласования функциональных требований должен происходить на модельном уровне, в нотации языка моделирования UML.

В рамках использования унифицированного процесса разработки ПО следует:

- понять природу и сферу применения поставленной задачи, в рамках соответствующего бизнеса, то есть, уяснить суть лабораторного задания;
- собрать все функциональные требования и провести их анализ;
- построить концептуальную модель предметной области (бизнеса);
- на базе контекста, определяемого концептуальной моделью, разработать статические и динамические модели анализа и специфицировать соответствующие артефакты;
- разработать прототип пользовательского интерфейса;
- выполнить тестирование полученных артефактов.

Содержание лабораторных занятий включает следующие темы:

Установление требований

Моделирование предметной области.

Моделирование системных вариантов использования.

Разработка динамических и статических моделей.

Разработка сущностного прототипа пользовательского интерфейса.

Валидация моделей вариантов использования и пользовательского интерфейса.

Планирование работ проекта

Управление проектом

Лабораторная работа №1 – Установление требований. Функциональное моделирование и построение моделей с помощью PLATINUM BPwin.

Цель работы - состоит в том, чтобы дать развернутое определение функциональных, а также нефункциональных требований, которые необходимо утвердить в системе для проектирования.

Методические указания к лабораторной работе

Требования определяют формулировки сервисов, ожидаемых от системы и формулировки ограничений, которым система должна подчиняться.

Формулировки сервисов объединяются в следующие группы:

- сервисы, описывающие границы системы;

-сервисы, описывающие бизнес-функции (*функциональные требования*);

-сервисы, описывающие требуемые структуры данных (*требования к данным*).

Формулировки ограничений можно классифицировать в соответствии с различными категориями ограничений, накладываемых на систему, такими как производительность, надежность и доступность, обработка ошибок, интерфейсные требования, ограничения на точность, инструменты и языки программирования и т.д.

Процесс выявления требований заключается в получении требований от пользователей и владельцев системы. ***Методы выявления требований***: интервью с заказчиком и построение программного прототипа с помощью которого раскрываются дополнительные требования.

Собранные требования должны быть подвергнуты тщательному анализу для выявления в них повторов и противоречий.

Документирование требований заключается в том, что им дают *определение, классифицируют, нумеруют и присваивают приоритеты.*

Кроме описательной части документирование требований должно включать высокоуровневую схематичную бизнес-модель, которая, как правило, состоит из *модели границ системы, модели бизнес-вариантов использования и модели бизнес-классов.*

Требования имеют тенденцию изменяться! Чтобы справиться с изменчивыми требованиями, необходимо уметь управлять изменениями. Упраление требованиями включает такие виды деятельности как оценка влияния изменений одних требований на другие, а также на неизменяемую часть системы.

Бизнес – модель требований. На этапе *установления требований* осуществляется выявление требований и их определение, преимущественно, в виде формулировок на *естественном языке*. Формальное моделирование требований с использованием языка UML проводится позже на этапе *спецификации требований*. Тем не менее, во время установления требований постоянно ведется деятельность по обобщенному визуальному представлению собранных требований, называемая *бизнес-моделированием требований.*

Понятие «рамки системы» связано с отслеживанием того, чтобы изменения заявленных требований не выходили за пределы принятых рамок проекта.

Для ответа на вопрос о *рамках системы*, необходимо знать. В каком контексте функционирует система. Небходимо знать, какие внешние сущности (другие системы, организации, люди и т.д.) рассчитывают на получение услуг от системы или готовы предоставить услуги нам. В бизнес – системах подобные услуги приобретают форму информации и представляются потоками данных. Поэтому рамки системы можно определить, обозначив внешние сущности и входные/выходные потоки данных между внешними сущностями и нашей системой. Система получает входную информацию и выполняет ее необходимую обработку с целью выработки выходной информации. Всякое требование, которое не может быть поддержано за счет внутрисистемных возможностей обработки, выходит за рамки системы.

Для установления рамок системы используются следующие модели: модель бизнес-вариантов использования и модель бизнес-классов.

Модель бизнес-вариантов использования. Данная модель связана с моделированием бизнес-процессов, то есть, с изучением деятельности организации, включающей: исследование структуры организации, роли сотрудников в этой структуре и взаимосвязи между сотрудниками; анализ рабочих потоков в организации – главных процессов в ее деятельности.

Моделирование бизнес-процессов, таким образом, устанавливает контекст для остальной части проекта, и помогает не забывать об основных побудительных причинах построения системы.

Модель бизнес-вариантов использования представляет собой модель вариантов использования на верхнем уровне абстракции. Модель бизнес-вариантов использования концентрируется на архитектуре бизнес-процессов и дает взглянуть на предполагаемое поведение системы так сказать « с высоты птичьего полета».

На этапе спецификации требований, бизнес-варианты использования превращаются в детальные варианты использования, включающие в себя подпроцессы и альтернативные процессы, некоторые копии экранов, демонстрирующие GUI -интерфейс, а также взаимосвязи между введенными вариантами использования. Внешние сущности заменяются субъектами. Субъекты активны. Они управляют процессом. Они активизируют варианты использования, отправляя им сообщения о событиях. Варианты использования управляются событиями. Линии, связывающие субъектов и вариантов использования, - это не потоки данных. Эти линии связи представляют *поток событий*, исходящих от субъектов и *поток откликов*, исходящих от вариантов использования.

Модель бизнес-классов. Модель бизнес-классов - это модель классов на высоком уровне абстракции. Она определяет основные «бизнес-объекты» системы – структуры данных бизнес-процессов, которые составляют основу системы и определяют ее деятельность. На этом уровне классы специфицируются именем и кратким описанием обязанностей. В качестве отношений между бизнес-классами достаточно отношений: ассоциация, наследование, агрегация и зависимость.

Спецификация требований. Требования необходимо специфицировать (то есть задать) графически или каким-либо иным формальным способом.

Спецификация – итеративный процесс с пошаговым наращиванием уровня детализации моделей спецификаций.

Спецификация требований связана с доскональным моделированием требований заказчиков, определенных в процессе установления требований.

Модели спецификаций разделяются на три группы.

- Модели состояний.
- Модели поведения.
- Модели изменения состояний.

Модели состояний – «детализируют» требования к данным. *Модели поведения* – обеспечивают детализированные спецификации для

функциональных требований. *Модели изменения состояний* охватывают два приведенных выше вида требований. Они призваны объяснить, каким образом действие функций приводит к изменению данных.

Каждая диаграмма дает представление об определенной стороне системы, а взаимосвязанный набор – о всей системе.

Концептуальная диаграмма предметной области определяет все три аспекта – *состояние, поведение объектов*, и, косвенно, *изменения состояний объектов*.

Спецификации состояний

Состояние объекта определяется значениями его атрибутов и ассоциаций. Поскольку состояния объекта определяются ***структурами данных, модели структур данных называются спецификациями состояний***

Моделирование состояний называют статическим моделированием. Основной задачей статического моделирования является определение *базовых сущностей (классов)* предметной области, а также их атрибутов и отношений с другими базовыми сущностями. Операции этих сущностей пока не рассматриваются, они выводятся из моделей спецификации поведения.

В типичной ситуации сначала определяются:

- базовые классы-сущности, которые определяют предметную область;
- классы, которые обслуживают системные события (управляющие классы) :
- классы, которые представляют GUI-интерфейс (такие классы не устанавливаются до тех пор, пока не станут известны все поведенческие характеристики системы).

Спецификация поведения

Поведение системы – это ее реакция в ответ на внешние события. В языке UML внешне наблюдаемое и допускающее тестирование поведение фиксируется в виде вариантов использования (ВИ).

Вариант использования – представляет собой некий целостный набор функций, имеющих определенную ценность для субъекта.

Вариант использования можно вывести в результате *идентификации задач для субъекта*. Для этого следует задаться вопросом: «Каковы обязанности субъекта по отношению к системе и чего он ожидает от системы?». Вариант использования также можно определить в результате непосредственного анализа функциональных требований. Во многих случаях функциональное требование отображается непосредственно в вариант использования.

На этапе анализа варианты использования вбирают в себя системные требования, концентрируясь на том, что делает или должна делать система. На этапе проектирования представление проектных решений в виде вариантов использования можно использовать для спецификации поведения системы в том виде, как оно должно быть реализовано.

Поведение системы, закрепленное с помощью вариантов использования. Можно смоделировать с помощью диаграмм видов деятельности, а взаимодействие объектов – с помощью диаграмм взаимодействий.

Спецификация поведения позволяет взглянуть на систему с точки зрения ее функционирования. Здесь основная задача состоит в том, чтобы определить варианты использования для области приложений и установить, какие классы – сущности участвуют в выполнении этих вариантов использования. При этом необходимо идентифицировать операции классов и сообщения, передаваемые между объектами.

Спецификация поведения дает функциональный взгляд на статическое состояние системы. Все изменения объектов, при их взаимодействии, находят выражение в спецификациях изменения состояний.

По мере создания моделей поведения появляются еще два уровня классов.

- классы, которые обслуживают события, инициируемые
- классы, представляющие GUI -интерфейсы (пограничные классы).

Спецификация изменения состояний. Ссостояние во времени, служат спецификация изменения состояний.

Состояние объекта определяется значениями его атрибутов, включая атрибуты отношения.

Спецификация состояний, помимо прочего, определяет атрибуты класса. Спецификация поведения определяет операции класса, некоторые из которых могут изменять состояние объекта.

Моделирование состояний объектов осуществляется с помощью диаграмм состояний. Граф состояний (автомат) – это граф состояний и переходов. Модели состояний строятся, как правило, для значимых, характерных для системы классов - сущностей.

Процесс исследования состояний объектов основан на анализе содержимого атрибутов классов и выявлении того, какие из этих атрибутов представляют особый интерес с точки зрения вариантов использования. Не все атрибуты влияют на изменения состояний. Изменение значения атрибута может представлять интерес как изменение состояния объекта, и это изменение может быть необходимо зафиксировать и отразить в диаграмме состояний.

Переходы между состояниями активизируются при появлении определенных событий или при выполнении определенных условий. Условие активизирует переход всякий раз, когда вид деятельности в состоянии завершен и условие принимает значение «истина».

Выполнение лабораторной работы заключается в анализе и разработке С – требований, согласно следующего плана :

Введение

1. Предварительные замечания к проекту

1.1. Цели и рамки проекта.

1.2. Идеи в отношении решений

Обсуждение идей, касающихся начальных проектных решений по разработке, либо использовании готовых решений, элементов и т.д.

2. Системные сервисы

2.1 Рамки системы.

Рамки системы можно моделировать с помощью *концептуальной модели предметной области.*

2.2. Функциональные требования.

Функциональные требования можно моделировать с помощью диаграммы бизнес-вариантов использования. При этом все функциональные требования необходимо обозначить, классифицировать и определить.

2.3. Требования к данным.

Требования к данным можно моделировать с помощью *диаграммы бизнес-классов или диаграммы сущность-связь*. При этом необходимо описать атрибутное наполнение классов и определить идентифицирующие атрибуты классов, для правильного представления ассоциаций между классами

3. Системные ограничения

Этот тип ограничений определяет, что должна делать система, насколько система ограничена при выполнении обслуживания, и связан со следующими видами требований:

1.1 Требования к интерфейсу.

Определяют, каким образом система будет взаимодействовать с пользователями. Начальное проектирование прототипа пользовательского интерфейса проводится во время спецификации требований и позже, во время системного проектирования.

1.2 Требования к производительности

Задают время отклика системы, с которой должны выполняться различные задания, а также включают такие ограничения, как: надежность, готовность, пропускную способность и т.д.

3.3 Требования к безопасности

Описывают пользовательские права доступа к информации, контролируемые системой, а также предоставление ограниченного доступа к данным и на выполнение определенных операций с данными.

3.4 Эксплуатационные требования

Определяют программно-техническую среду, если она известна на этапе проектирования, в которой должна функционировать система.

3.5 Другие ограничения

Например, в отношении некоторых систем могут предъявляться повышенные требования к легкости их использования, либо к легкости их сопровождения.

4. Проектные вопросы

4.1 Открытые вопросы

Здесь поднимаются вопросы, которые могут сказаться на успехе проекта и которые не рассматривались в других разделах документа. Сюда относится ожидаемое возрастание значения некоторых требовании , которые пока

выходят за рамки проекта, а также любые потенциальные проблемы и отклонения в поведении системы, которые могут начаться в связи с развертыванием системы.

1.2 Предварительный план-график

Включает план-график выполнения основных проектных заданий и распределение необходимых людских и других ресурсов.

1.3. Оценка стоимости проекта

Прямым результатом составления плана – графика может быть разработка ранней оценки стоимости и сроков выполнения проекта с использованием функционально – ориентированных метрик.

Приложения

Содержат остальную, полезную для понимания требований информацию.

Глоссарий

Термины, сокращения и аббревиатуры, используемые в документе описания требований.

Деловые документы и формы

Заполненные документы и формы, используемые в процессах делопроизводства.

Ссылки

Ссылки содержат перечень документов, используемые при подготовке документа описания требований: книги, электронные адреса, протоколы совещаний, служебные записки и внутренние документы.

Основная литература: 1, 3,5,7

Контрольные вопросы:

1. Как с помощью вариантов использования и таблиц «событие - реакция» зафиксировать пользовательские требования.

2. Перечислите все методы выявления требований, использовавшиеся вами для текущего проекта.

3. Определите способы выявления требований, которые, по-вашему, дадут лучший результат.

4. Так как выявить требования полностью невозможно, поэтому существуют признаки того, что источники сведений уже почти иссякли. Какие это признаки?

5. Каких ловушек следует опасаться при установлении требований с применением вариантов использования?

Лабораторная работа №2 – Построение диаграммы IDEF. Построение диаграммы IDEF3. Моделирование предметной области

Цель работы – рассмотреть методы изучения бизнеса, его составные части и рабочие потоки, для того, чтобы понять принципы ведения бизнеса перед разработкой программной (информационной) системы. Система обязана соответствовать уникальным особенностям конкретного бизнеса.

Методические указания к лабораторной работе

Моделирование предметной области включает:

- моделирование бизнес-процессов;
- спецификацию бизнес-правил;
- учет внешних сущностей, имеющих отношение к бизнесу;
- моделирование бизнес-сущностей предметной области.

Моделирование бизнес-процессов предполагает исследование внутренних и внешних компонентов бизнеса и их взаимосвязи. В UML для документирования полученных данных используется бизнес-модель.

Моделирование бизнес-процессов является первым этапом унифицированного процесса разработки, результатом которого является установление контекста для остальной части проекта. Во время проектирования системы, моделирование бизнес-процессов помогает не забывать об основных побудительных причинах построения системы.

В основе концепции моделирования бизнес-процессов входят такие понятия, как: бизнес-роль, сотрудник, Бизнес-вариант использования, диаграмма Бизнес-вариантов использования, Бизнес-сущность и диаграмма Деятельности.

Бизнес-роль исполняет человек или подразделение, внешнее по отношению к организации, но взаимодействующие с ней.

Сотрудник – роль внутри организации (не должность). Один человек может играть несколько ролей, занимая одну должность.

Для моделирования сотрудников необходимо рассмотреть следующие вопросы:

- каковы ответственности сотрудника?
- как он взаимосвязан с другими сотрудниками ?
- в каких рабочих потоках он участвует?
- каковы обязанности сотрудника в каждом из рабочих потоков?
- какую квалификацию должен иметь сотрудник для исполнения своей бизнес-роли? (Этот вопрос важен также с точки зрения проектирования пользовательского интерфейса).

Бизнес- вариант использования - это группа связанных рабочих потоков внутри организации, предоставляющая данные для бизнес-роли.

Бизнес- вариант использования описывает деятельность организации, точнее то, что делается в организации для ведения бизнеса и тех, кто ведет этот бизнес. Набор всех бизнес-вариантов использования для организации должен полностью описывать бизнес этой организации.

Примерами бизнес-вариантов использования для электронного бизнеса могут служить: «Зарегистрировать нового пользователя», «Создать/Изменить заказ», «Пополнить склад», «Отменить заказ». Для инвестиционной компании: «Купить акции», «Продать акции». Для высшего учебного заведения: «Выпуск учебно-методической литературы», « Запись на университетские курсы», «Формирование учебных планов» и т.д.

Обычно названия бизнес -вариантов использования имеют формат <глагол><существительное>, например «Назначить цену». Использование такого стандарта связано как с унификацией названий для всех

разработчиков, а также с тем, что такие названия проще понять, и, что важнее всего, название сконцентрировано на бизнесе, то есть на исполняемой операции, а не сущности.

Для любого бизнес-варианта использования требуется, руководствуясь бизнес-правилами, сформировать некоторое описание, позволяющее понять, что и каким образом выполняется в таком варианте. Ответы на вопросы о том, что и каким образом выполняется не известны, пока не документирован определенный рабочий поток в организации.

Документом рабочего потока является нумерованный пошаговый список того, что происходит во время прохода через бизнес-вариант использования. Например, для магазина розничной торговли, документирование рабочего потока будет выглядеть следующим образом:

1. ***Клерк*** *запрашивает* у ***менеджера список*** всех ***новых товаров***, для которых требуется назначить цены.
2. Клерк *изучает* ***закупочную ведомость***, чтобы определить закупочную стоимость каждого ***товара.***
3. Клерк *начисляет* 10% к закупочной цене, чтобы определить цену товара.
4. Клерк *отправляет* ***список новых цен*** на утверждение менеджеру.
5. Если менеджер *не утверждает* цену, то совместно с клерком он должен *изменить* предложенную цену товара.
6. Клерк *формирует* ***ведомость цен*** для всех новых товаров.
7. Клерк *проставляет* в ведомости цен цены на все новые товары.

Выделенные курсивом слова, определяют бизнес-процессы, происходящие в бизнес-варианте использования, в соответствии с бизнес-правилами, принятыми в организации. Такие глаголы в дальнейшем будут играть роль событий в системных моделях анализа и проектирования.

Выделенные курсивом и жирно слова, являются сотрудниками и бизнес-сущностями предметной области. Бизнес-сущности – это проблемные понятия, принятые и употребляемые в организации. Такие понятия составляют словарь терминов (существительные) предметной области. Такие понятия можно рассматривать как классы анализа и как способы организации требований. В дальнейшем бизнес-сущности будут представлять ядро объектной модели проектируемой системы.

Диаграмма Бизнес-вариантов использования

На диаграммах этого типа показаны Бизнес-варианты использования, бизнес-роли и сотрудники организации, а также отношения между ними. Это дает полную модель хозяйственной деятельности организации и описание внешних по отношению к ней сущностей. Диаграмма Бизнес-вариантов использования описывает зону действия организации, то есть ее окружение и границы.

Диаграмма предназначена для быстрого изучения высокоуровневых отношений в бизнесе без показа всех подробностей.

Диаграмма деятельности

Диаграмма деятельности – это способ графического изображения модели рабочих потоков для вариантов использования. Диаграмма показывает шаги (этапы) рабочего потока, точки принятия решений в этом потоке, ответственности для каждого этапа и объекты, влияющие на рабочий поток.

Основными элементами диаграммы деятельности являются:

- *Дорожки*, показывающие сферы ответственности сотрудников, участвующих в Бизнес-варианте использования.
- *Деятельности* (activity), демонстрирующие этапы рабочих потоков.
- *Действия* (actions), показывающие шаги в пределах каждой деятельности. Действие может происходить при входе в деятельность, во время ее исполнения, при выходе из деятельности или при возникновении определенного события.
- *Бизнес-объекты*, являющиеся сущностями, воздействующими на рабочий поток. На эти объекты воздействует рабочий поток, в результате чего объекты изменяют свое состояние.
- *Переходы,* показывающие движение рабочего потока между деятельностями.
- *Точки принятия решений*, демонстрирующие решения, которые необходимо принять в пределах рабочего потока.
- *Синхронизации*, иллюстрирующие два или более одновременных этапа в пределах рабочего потока.
- *Исходное состояние*, определяющее начало рабочего потока.
- *Конечное состояние*, определяющее завершение рабочего потока.

Порядок выполнения лабораторной работы

1. В соответствии с выбранной темой, следует определить границы, в которых моделируется бизнес и рассмотреть рабочие потоки. Для этих целей будет полезно использовать такие известные средства методологии структурного анализа, как DFD (Data Flow Diagrams -диаграммы потоков данных совместно со словарями данных и спецификациями процессов), SADT (Structured Analysis and Design Technique -технология структурного анализа и проектирования) [17].

2. Произвести идентификацию бизнес-ролей, бизнес-вариантов использования и сотрудников. Чтобы выявить бизнес-роли анализируется область действия проекта и то, что находится вне его – что находится вне организации, но связано с ней, то есть *кто и что* взаимодействует с бизнесом.

Для идентификации сотрудников организации следует проанализировать рамки проекта и определить всех критичных участников бизнеса и их роли, основываясь на должностях участников.

После выявления сотрудников необходимо детализировать их обязанности и взаимодействия с другими бизнес-ролями и сотрудниками. Для идентификации бизнес-вариантов использования следует начать с анализа общей концепции бизнеса, то есть с особенностей бизнеса по отношению к внешнему миру. Далее следует идентифицировать основные

рабочие потоки в бизнесе организации, связанные со всеми аспектами бизнеса.

3. Следующий этап – формирование одной или более диаграмм Бизнес-вариантов использования, показывающих отношения взаимодействия между сотрудниками, бизнес-ролями и бизнес-вариантами использования.

4. Документирование подробностей.

Для каждого бизнес-варианта использования через рабочий поток идентифицируется и документируется один или более вариантов использования. Для документирования рабочего потока используются нумерованные списки, графики производственных процессов или диаграммы деятельности. При этом, как правило, рабочий поток должен кроме основного потока, выполняемого в обычных условиях, содержать и все его альтернативные потоки.

Таким образом, варианты использования показывают деятельность организации, рабочие потоки определяют подробности выполнения вариантов использования, роли описывают связанные с организацией внешние объекты, а сотрудники иллюстрируют работников организации, с которыми взаимодействуют внешние роли. Документ о бизнес-вариантах использования предоставляет дополнительную информацию о вариантах использования.

Основная литература: 1,3,5,7.

Контрольные вопросы:

1. Какие методы моделирования применяются при структурном подходе к разработке систем?
2. Каково различие между бизнес-процессом и бизнес-функцией? В чем заключается это различие? Приведите пример бизнес-процесса.
3. Какими особенностями к анализу и проектированию отличается структурный подход?
4. Дайте определение бизнес-правила и приведите примеры.
5. Что такое бизнес-сущности? Приведите примеры бизнеса с бизнес-сущностями и дополните их атрибутами и операциями.
6. Рассмотрите пример с магазином розничной торговли, выявите все бизнес- сущности и промоделируйте их взаимосвязи методом CRC - карточек.

Лабораторная работа №3 – Построение диаграммы DFD. Проектирование моделей данных с помощью ErWin. Моделирование вариантов использования

Цель работы – освоить основные фундаментальные концепции на основе моделирования вариантов использования (системного моделирования).

Методические указания к лабораторной работе

Моделирование системных вариантов использования включает:

- выявление действующих лиц и инициируемых ими событий;

- описание сервисов, предоставляемых системой и идентификация вариантов использования;

- прослеживание вариантов системного использования до бизнес-вариантов использования;

- прослеживание требований до вариантов системного использования;

- спецификацию системных вариантов использования (в нотации Г.Буча, либо А.Кокберна);

- построение диаграмм вариантов использования.

Если в бизнес-моделировании акцент делается на саму организацию, тогда как в системном моделировании основное внимание нацелено на разрабатываемую систему. Вариант использования в системном моделировании описывает действия системы в контексте бизнеса; действующее лицо – это внешнее по отношению к системе; сотрудник – не используется.

Действующее лицо (субъект) - некто или нечто, взаимодействующее с вариантом использования. Субъект взаимодействует с вариантом использования инициируя события и ожидая получить некий полезный результат.

Варианты использования описывают все, что попадает в зону действия системы, а действующие лица – все, что находится вне границ действия системы: пользователи системы, другие системы и время.

Присваивая имена действующим лицам, необходимо отразить роль, а не должность, так как каждый человек может играть несколько ролей.

Поведение системы - это ее реакция в ответ на внешние события. В языке UML внешне наблюдаемое и допускающее тестирование поведение системы фиксируется в виде вариантов использования.

Вариант использования (Use Case) выполняет бизнес-функцию, которую может наблюдать действуюшее лицо и котрая может быть впоследствии отдельно протестирована в процессе разработки.

Вариант использования представляет собой некий целостный набор функций, имеющий определенную ценность для субъекта.

Субъекты и варианты использования определяются в результате анализа функциональных требований. Функциональные требования воплощаются (отображаются) в вариантах использования, а последние удовлетворяют функциональные требования за счет предоставления субъекту полезного результата.

Варианты использования можно вывести в результате идентификации задач, при анализе бизнес-вариантов использования. Каковы обязанности субъекта по отношению к системе и чего он ожидает от системы?

Разделение проекта по вариантам использования ориентировано на процессы в системе, то есть на том, что должна делать система, а не на их реализацию.

Каждый вариант использования в системе должен прослеживаться до соответствующего бизнес-варианта использования. *Системный вариант использования реализует часть функций бизнес-варианта использования*.

После прослеживания вариантов системного использования до бизнес-вариантов использования, следует отследить функциональные требования до вариантов системного использования.

Спецификация системных вариантов использования

Варианты использования описывают то, что будет делать система. Однако для реального проектирования системы потребуется более специфические данные, которые отражены в потоке событий.

Поток событий подробно определяет то, что будет делать с системой пользователь, и то, что делает сама система.

Обычно поток событий содержит: название и краткое описание варианта использования, с какими субъектами работает вариант использования, предусловия, основной поток событий, альтернативные потоки событий, постусловия.

Описание - краткое объяснение действий в варианте использования.

Предусловия - все условия, которые должны быть выполнены прежде, чем данный вариант начнет работать.

Основной и альтернативные потоки событий содержат:

- процедуру начала варианта использования;
- различные этапы исполнения варианта использования;
- нормальный (основной) поток событий в варианте использования, который описывает наилучший сценарий исполнения варианта использования;
- все отклонения от основного потока событий –альтернативные потоки событий и которые не рассматриваюся как ошибочные.
- все потоки ошибок. Потоком ошибок считается отклонение от основного или альтернативного потока, которое рассматривается как условие формирования ошибки. Поток ошибок опысывает проблемы в самой системе, а не ошибки задачи.
- *Постусловия* – это условия, которые всегда равны true после завершения исполнения варианта использования. Как и предусловия, постусловия могут добавлять информацию о порядке исполнения вариантов использования. Например, если один вариант исполняется всегда после другого, то это можно отразить в постусловии. Постусловия требуются не для всех вариантов использования.

Таким образом, моделирование бизнес-вариантов представляет *контекст* разрабатываемой системы, а моделирование системных вариантов использования – *архитектуру системы*, полный набор функциональных возможностей системы. Такое представление должно связать действующих лиц, функциональные требования, варианты использования и классы-сущности задачи. Для этих целей целесообразно строить следующие таблицы[4].

Таблица 1 – Распределение требований по субъектам и вариантам использования.

Таблица 2 – Описательная спецификация варианта использования

Таблица 3 - Соответствие функциональных требований и классов-сущностей.

Выделение классов-сущностей представляет собой итеративную задачу. При определении того, являются ли понятия, присутствующие в требованиях, искомыми классами, могут помочь ответы на следующие вопросы.

1. Является ли понятие «вместилищем» данных ?
2. Обладает ли оно отдельными атрибутами, способными принимать разные значения?
3. Можно ли создать для него множество объектов-экземпляров?
4. Входит ли оно в границы прикладной области?

Диаграмма Вариантов использования

Диаграмма Вариантов использования показывает некоторые варианты использования в системе некоторых действующих лиц и отношения между ними. Предоставляется высокоуровневое описание системы, причем действующим лицом становится все и все, что взаимодействует с разрабатываемой системой. На диаграмме видны системные действующие лица, системные варианты использования и отношения между ними. Полный набор вариантов использования и действующих лиц покажет границы проекта, что позволит выявить все упущенные в нем функции.

Приведем несколько рекомендаций для создания диаграмм вариантов использования.

- Не моделируйте связи между действующими лицами (хотя допустимо их обобщение). Для связи между ними служат диаграммы рабочих потоков.
- Не моделируйте прямые связи между двумя вариантами использования (хотя допустимо моделирование включающих «<< include>>» и расширяющих «<<extend>>» отношений). Диаграммы не документируют порядок выполнения вариантов использования, поэтому не требуется указывать между ними ассоциации.
- Стрелки должны быть направлены от действующего лица к варианту использования. Исключением являются включающие и расширяющие отношения.
- Считайте базу данных уровнем, находящимся ниже всей диаграммы. Можно ввести информацию в базу данных в одном варианте использования, а затем получить эту информацию в другом. В этом случае не нужно показывать информационный поток между этими двумя вариантами использования.

Порядок выполнения лабораторной работы

1. Произвести идентификацию действующих лиц и инициируемые ими события, в соответствии с ролями, которые они выполняют в задаче.
2. Идентифицировать все системные варианты использования и произвести их прослеживание до бизнес-вариантов использования, а также прослеживание требований до выявленных вариантов системного использования.
3. Построить *Таблицы* 1,2,3.

4. Построить диаграмму Вариантов использования

Основная литература: 1,2,3,7.

Контрольные вопросы:

1. Что такое вариант использования? Как и когда стартует и заканчивается вариант использования?
2. Когда вариант использования взаимодействует с действующими лицами и в чем заключается характер взаимодействия?
3. Спецификация варианта использования в нотации Г.Буча и в нотации А.Кокберна. Каковы отличия этих нотаций?
4. В чем отличие основного потока событий от альтернативных, и как они отображаются на диаграмме Вариантов использования?
5. Каковы типы отношений между вариантами использования, и в чем логика этих отношений?
6. Что такое классы-сущности?

Лабораторная работа №4 - Структурное проектирование. Методология SADT. Динамические и статические контентные модели

Цель работы- усвоение того, каким образом функциональные требования и требования к данным, выявленные на этапе установления требований, связаны с моделями состояний, поведения и изменения состояний, разрабатываемых на этапе спецификации требований.

Методические указания к лабораторной работе

Суть контентных моделей проекта заключается в спецификации требований. В результате спецификации требований вырабатываются три категории моделей: *модели состояний, модели поведения и модели изменения состояний*. Модели представляются в виде диаграмм на языке визуального моделирования UML.

Спецификация состояний описывает проект со статической точки зрения классов, содержания их атрибутов и их отношений.

Спецификация поведения описывает проект с операционной точки зрения. Движущей силой спецификации поведения и, конечно, анализа требований и проектирования системы в целом, выступают варианты использования. Диаграммы вариантов использования дают только наглядное представление – истинная сила вариантов использования заключается в их неформальных спецификациях. Остальные поведенческие диаграммы являются производными от моделей вариантов использования. Они включают диаграммы видов деятельности, диаграммы взаимодействия и введение операций в классы.

Спецификация изменения состояний описывает проект с динамической точки зрения. Объекты «бомбардируются» событиями ,и некоторые из этих событий вызывают изменения состояний объектов. Диаграммы состояний позволяют моделировать изменения этих состояний.

Статические модели

Статические модели - обеспечивают представление структуры системы в терминах классов и отношений между ними. Основным средством для представления статических моделей являются диаграммы классов.

В ходе анализа диаграммы классов используются для указания ролей и обязанностей сущностей, которые обеспечивают поведение системы.

В ходе проектирования Диаграммы классов используются для фиксации структуры классов, которые формируют системную архитектуру.

Вершины Диаграмм классов нагружены классами, а дуги (ребра) – отношениями между ними.

Спецификация класса содержит: имя класса, свойства (атрибуты) и операции.

Синтаксис представления *свойства* класса имеет вид:

Видимость Имя[Множественность]:Тип = НачальЗначение {Характеристики}

Видимость : public , private, protected, internal.

Характеристики: *changeable* - нет ограничений на модификацию значения свойства; *frozen* - после инициализации объекта значение свойства не изменяется и т.д.

Синтаксис представления операции имеет вид

Видимость Имя (Список Параметров) : ВозвращаемыйТип {Характеристики}

Формат представления параметра имеет следующий синтаксис:

Направление Имя : Тип = ЗначениеПоУмолчанию

Направление : in - входной параметр, не может модифицироваться; out - выходной параметр, может модифицироваться для передачи информации в вызывающий объект; inout - входной параметр, может модифицироваться.

Характеристики : leaf - конечная , не полиморфная операция; isQuery - выполнение операции не изменяет состояния объекта и т.д.

Для ограничения количества экземпляров класса, либо свойств класса используется свойство Множественность. Множественность свойства задается выражением в квадратных скобках после имени свойства, количество экземпляров класса – в правом верхнем углу прямоугольника, изображающего класс.

Отношения в диаграммах классов

Отношения между классами показывают:

- *ассоциации* - структурные отношения вместе с метками, навигацией, ролями и их мощностями, видимостью. Разновидностями структурных отношений являются отношения агрегации и композиции;
- *зависимости* – отношения использования;
- *обобщения* – отношения между общим предметом и его специализированной разновидностью;
- *реализация* - семантическое отношение между классами, в котором класс –приемник выполняет реализацию операций интерфейса класса – источника.

Динамические модели

Динамические модели - обеспечивают представление поведения систем, то есть отражают изменение состояний в процессе работы системы.

Для моделирования поведения системы используют *автоматы и взаимодействия.*

Автомат – описывает поведение системы в терминах последовательности состояний, через которые проходит реактивный объект, в течение своей жизни, реагируя на события, - в том числе описание реакций на эти события.

Взаимодействие – описывает поведение системы в терминах обмена сообщениями между объектами.

Диаграмма состояний

Автоматы отображают с помощью диаграмм схем состояний (диаграммы состояний), которые можно присоединять к классам, вариантам использования или к системе в целом.

Состояние – это ситуация в жизни объекта, на протяжении которой он удовлетворяет некоторому условию, осуществляет определенную деятельность или ожидает какого-то события.

Событие - это стимул, способный вызвать срабатывание перехода.

Переход - это отношение между двумя состояниями, показывающее, что объект, находящийся в первом состоянии, должен выполнить некоторые действия и перейти во второе состояние, как только произойдет определенное событие и будут выполнены заданные условия.

Деятельность – это продолжающееся неатомарное вычисление внутри автомата.

Действие - это атомарное вычисление, которое приводит к смене состояния или возврату значения.

Диаграмма состояний изображается в виде графа с вершинами и ребрами. В них основное внимание уделяется переходам из состояния в состояние, то есть переходам потока управления от одного состояния к другому, а не деятельности к деятельности (последнее можно выполнить с помощью блок-схем или диаграмм деятельности).

Таким образом, при моделировании поведения реактивного объекта нужно специфицировать главным образом три вещи:

- устойчивые состояния, в которых может находиться объект;
- события, которые инициируют переходы объекта из одного состояния в другое;
- действия, выполняемые при каждой смене состояния объекта.

Диаграммы взаимодействий

На диаграммах взаимодействий показывают связи, включающие множество объектов и отношений между ними, в том числе сообщения, которыми объекты обмениваются. При этом диаграмма последовательностей акцентирует внимание на временной упорядоченности сообщений, а диаграмма кооперации – на структурной организации посылающих и принимающих сообщения объектов.

Диаграммы последовательностей визуализируют и специфицируют основные и альтернативные сценарии, описывающие взаимодействие определенных объектов и сообщения, которыми они обмениваются.

Этапами создания диаграмм взаимодействия являются: поиск объектов, поиск действующих лиц и добавление сообщений в диаграмму.

Поиск объектов лежит в области спецификации вариантов использования, где рассматриваются сценарии прохождения потока событий. Выявление объектов осуществляется по таким категориям, как объекты-сущности, граничные объекты и управляющие объекты.

Граничные объекты формируют интерфейс системы, а *управляющие объекты* координируют и управляют другими объектами в общей логике потока событий.

Действующие лица выделяются во время анализа потока событий как сущности, запускающие процессы, в рамках определенного сценария.

Диаграммы взаимодействия – краеугольный камень, на котором возводится оставшаяся часть проекта, так как они позволяют:

- определять классы, которые нужно еще создать;

-определять и уточнять связи между классами;

- определять и уточнять операции и ответственности каждого класса.

Диаграммы последовательности полезны для того, чтобы понять логическую последовательность событий в сценарии. Диаграммы последовательности – это основной инструмент для принятия решений о распределении поведения.

Кооперативные диаграммы полезны в тех случаях, когда нужно оценить последствия сделанных изменений в объекте и на какие другие объекты это повлияет.

Создавая диаграммы Взаимодействия, следует помнить, что таким образом объектам назначаются определенные ответственности. Нужно следить за тем, чтобы объекты и их ответственности соответствовали друг другу. Например, экраны и формы обеспечивают только ввод и просмотр информации, и тогда внесение изменений в бизнес-логику не затронет интерфейс.

Хорошим способом анализа ответственностей является разделение всех объектов на категории сущности, граничные и управляющие.

Содержанием лабораторной работы - будет построение диаграмм последовательности для основных системных вариантов использования, диаграмм состояний для ключевых реактивных объектов, и в завершении, – диаграммы классов системы.

Порядок выполнения лабораторной работы

1. Скопируйте текст варианта использования из спецификации варианта использования и вставьте его в левое поле страницы, где будет происходить моделирование поведения объектов в сценариях данного варианта использования. В результате, требования к поведению системы будут находиться у вас перед глазами на протяжении всего процесса проектирования.

2. Добавьте на диаграмму действующие лица, граничные и сущностные объекты.

3. Определите, какие операции и в какие классы поместить – это суть моделирования взаимодействий. *Распределить поведение – это преобразовать управляющие объекты в операции и сообщения.*

При моделировании взаимодействий между различными объектами, большую помощь могут оказать стандартные паттерны (шаблоны), для стандартизации решения задач, возникающих в нескольких вариантах использования.

4. Обратиться к исходной диаграмме классов из модели предметной области, и отразить на ней все найденные проектные решения для данного варианта использования, после чего нарисовать соответствующую диаграмму последовательности. На практике моделирование классов и моделирование вариантов использования проводятся в параллель. Обе модели взаимно дополняют одна другую вспомогательной информацией.

5. По мере построения локальных диаграмм классов для каждого варианта использования, обновляйте статическую объектную модель системы, добавляя в нее каждый раз объекты из пространства задачи и из пространства решения (инфраструктурные классы).

6. Заключительным шагом выполнения лабораторной работы является - рецензирование окончательного проекта, при котором следует удостовериться в том, что все ответы на вопрос *как*, найденные в ходе детального проектирования, отражены на диаграммах последовательности и ассоциированных диаграммах классов (статической модели проекта) и согласуются с тем, *что* специфицировано вариантами использования.

Основная литература: 1,2,3,4,5].

Контрольные вопросы:

1. Поясните назначение статических моделей объектно-ориентированных программных систем?
2. Как используются статические модели?
3. Поясните общий синтаксис представления свойства (операции).
4. В чем смысл отношений зависимости и реализации?
5. Поясните два подхода к моделированию поведения системы. Объясните достоинства и недостатки каждого из этих подходов.
6. Охарактеризуйте средства и возможности диаграммы деятельности.
7. Когда удобнее применять диаграммы последовательности?
8. Что такое сценарий варианта использования?

Лабораторная работа №5 – Объектно-ориентированное моделирование и построение моделей с помощью объектно-ориентированной CASE-системы (Rational Rose). Проектирование интерфейса

Цель работы – заключается в выработке графических представлений пользовательского интерфейса, в соответствии с технологией, на которой базируется GUI-интерфейс.

Методические указания к лабораторной работе

Пользовательский интерфейс представляет собой совокупность программных и аппаратных средств, обеспечивающих взаимодействие пользователя с компьютером. Основу такого взаимодействия составляют диалоги.

Диалог – это регламентированный обмен информацией между пользователем и системой.

Обмен информацией осуществляется передачей сообщений и управляющих сигналов.

Сообщение – порция информации, участвующая в диалоговом обмене: запрос информации, запрос помощи, запрос операции или функции, ввод или изменение информации, выбор поля кадра и т.д.

Объектно-ориентированные интерфейсы используют модель взаимодействия с пользователем, ориентированную на манипулирование объектами предметной области.

Задача пользователя формулируется как целенаправленное изменение некоторого объекта, имеющего внутреннюю структуру, определенное содержание и внешнее символьное или графическое представление.

Пользователю предоставляется возможность:

- создавать объекты;
- изменять их параметры и связи с другими объектами;
- инициировать взаимодействие объектов.

Интерфейсы со свободной навигацией называют графическими пользовательскими интерфейсами (GUI –Graphic User Interface) . Графические интерфейсы поддерживают концепцию интерактивного взаимодействия с программным обеспечением, осуществляя визуальную обратную связь с пользователем и возможность прямого манипулирования объектами и информацией на экране. Кроме того, интерфейсы данного типа поддерживают концепцию совместимости программ, позволяя перемещать между ними информацию (технология OLE).

Существенной особенностью графических интерфейсов является способность изменяться в процессе взаимодействия с пользователем, предлагая выбор только тех операций, которые имеют смысл в конкретной ситуации. Реализуют такие интерфейсы, используя событийное программирование и объектно-ориентированные библиотеки, что предполагает применение визуальных сред разработки программного обеспечения.

Процесс проектирования пользовательского интерфейса сводится к определению множества необходимых диалогов, их основных сообщений и возможных сценариев использования системы.

Кроме сценариев необходимо использовать диаграммы состояний интерфейса или графы диалога.

Граф диалога – ориентированный взвешенный граф, каждой вершине которого сопоставлена конкретная картинка на экране (кадр), или определенное состояние диалога, характеризующееся набором доступных

пользователю действий. Дуги, исходящие из вершин, показывают возможные изменения состояний при выполнении пользователем указанных действий. В качестве весов дуг указывают условия переходов из состояния в состояние и операции, выполняемые во время перехода.

Таким образом, каждый маршрут на графе соответствует возможному варианту диалога. Причем представление диалога в виде графа в зависимости от стадии разработки может выполняться с разной степенью детализации.

Граф диалога - это граф состояний конечного автомата (машина Мура), моделирующего поведение системы при воздействиях пользователя. Для представления таких графов используется нотация *диаграмм состояний* UML .

В лабораторной работе речь идет о разработке прототипа пользовательского интерфейса, что означает разработку графа диалога верхнего уровня. Диалог на верхнем уровне должен обеспечивать реализацию диаграммы вариантов использования, поэтому исходный вариант графа диалога строим на основе анализа этой диаграммы.

Анализ системных вариантов использования диаграммы заключается в идентификации всех операций (сообщений), выделении и группировании среди них общих операций, сопоставлении с операциями условий их выполнения. Таким образом, для каждого варианта использования выявляются сценарии потоков событий, на базе которых разрабатывается граф диалога (карта диалога). Граф диалога представляет дизайн пользовательского интерфейса на высоком уровне абстракции. На нем показаны элементы диалога в системе и навигация между ними, но не показан подробный вид экрана. *Граф диалогов позволяет рассмотреть возможные концепции пользовательского интерфейса с учетом степени понимания требований к системе*.

Граф диалогов отражает сущность взаимодействия системы и пользователя,и основные моменты решения задачи. С помощью графа можно отследить отсутствующие, неправильные или ненужные переходы и, следовательно, отсутствующие, неправильные или ненужные требования.

Абстрактный концептуальный граф диалогов, разработанный в ходе анализа требований, становится руководством для подробной разработки пользовательского интерфейса.

Далее, необходимо определить, какие формы диалога можно использовать для каждого шага диалога: табличную, директивную, либо фразовую формы.

Последний этап проектирования интерфейса – разработка конкретных операций ввода/вывода для каждого диалога с учетом специфики формы диалога.

Рассмотрение и выбор интерфейсных компонентов, которые могут быть использованы в современных пользовательских интерфейсах, в лабораторной работе не предполагается, так как на этом этапе речь идет об абстракных диалогах для прототипа пользовательского интерфейса.

Результатом моделирования интерфейса будут являться модели:

-сущностного прототипа пользовательского интерфейса, отражающего требования к содержанию экрана/страницы;

-диаграммы потоков пользовательского интерфейса, описывающей навигацию потоков данных между сущностями прототипа.

Основная литература: 1,2,3,4,5.

Контрольные вопросы:

1. Назовите основные типы интерфейсов. Чем характеризуется каждый из них? Какими средствами реализуется? Какие типы интерфейсов являются основными в наше время?
2. Что понимают под термином «диалог»? Сколько диалогов может реализовывать система?
3. Назовите основные типы диалога и его формы. Какие модели используют для описания диалогов? Что служит исходными данными для проектирования диалогов?
4. Постройте граф диалога для простейшего графического редактора. Почему он имеет такой вид?
5. Какими свойствами обладает диаграмма состояний и что она включает в себя?

Лабораторная работа № 6 – Построение диаграммы прецедентов, взаимодействия и сотрудничества. Тестирование системных сервисов

Цель работы – рассмотреть и освоить методы тестирования с использованием моделей анализа и проектирования, а также объектно-ориентированное тестирование правильности.

Методические указания к лабораторной работе

Результатом проектирования является разработка следующих артефактов анализа и проектирования:

- Диаграмма вариантов использования.
- Диаграммы видов деятельности.
- Диаграммы взаимодействия.
- Диаграммы состояний.
- Диаграмма классов.
- Прототипа пользовательского интерфейса.

Построение прототипа пользовательского интерфейса можно отнести к объектно-ориентированному тестированию правильности. Подтверждение правильности объектно-ориентированного программного обеспечения ориентировано на видимые действия пользователя и распознаваемые пользователем выводы из системы.

Для упрощения разработки тестов правильности используются варианты использования, являющиеся частью модели требований. Каждый вариант использования задает сценарий, который позволяет обнаруживать ошибки во взаимодействии пользователя с системой. Для подтверждения правильности может проводиться обычное тестирование методом «черного ящика».

Полезную для формирования тестов правильности информацию содержат диаграммы взаимодействия, диаграммы видов деятельности, а также диаграммы состояний.

Следующим этапом выполнения лабораторной работы, будет проверка таких критериев тестирования, как правильность, полнота и согласованность.

Тестирование системных сервисов

Тестирование системных сервисов и управление изменениями – охватывает весь жизненный цикл разработки системы. Артефакты разработки каждого этапа должны быть протестированы. *Управление изменениями* – фундаментальный аспект управления проектом, заключающийся в документировании запросов на изменения и в отслеживании влияния изменений на артефакты разработки.

Тестирование GUI -интерфейса пронизывает весь процесс разработки ПО. Оно начинается на ранней стадии этапа выработки требований, в том числе, при разработке прототипов GUI –интерфейса. Такие ранние тесты GUI-интерфейса концентрируются на удовлетворении функциональных требований и удобстве использования приложения.

Разработка объектно-ориентированного ПО начинается с создания визуальных моделей, отражающих статические и динамические характеристики будущей системы. Вначале эти модели фикисируют исходные требования заказчика, затем формализуют реализацию этих требований путем выделения объектов, которые взаимодействуют друг с другом посредством передачи сообщений.

Исследование моделей взаимодействия приводит к построению *моделей классов* и их отношений, составляющих основу *логического представления системы*.

При переходе к *физическому представлению* строятся модели компоновки и размещения системы. На конструирование моделей приходится большая часть затрат объектно-ориентированного процесса разработки. Если к этому добавить, что цена устранения ошибки стремительно растет с каждой итерацией разработки, то совершенно логично требование тестировать объектно-ориентированные модели анализа и проектирования.

Критерии тестирования моделей

К критериям тестирования моделей относятся: *правильность, полнота и согласованность*.

- о ***синтаксической правильности*** судят по правильности использования языка объектного моделирования UML;
- о ***семантической правильности*** судят по соответствию модели реальным проблемам.

Для определения того, отражает ли модель реальный мир, она должна быть проанализирована на содержание классов, наследование классов, на выявление пропусков и неоднозначностей. Проверяется соответствие отношений классов реалиям физического мира.

О согласованности судят путем рассмотрения противоречий между элементами в модели. Несогласованная модель имеет в одной части представления, которые противоречат представлениям в других частях модели. Для оценки согласованности нужно исследовать каждый класс и его взаимодействия с другими классами. Для упрощения такого исследования удобно использовать модель Класс-Обязанность-Сотрудничество CRC (Class-Responsibility-Collaboration). Основной элемент этой модели - CRC - карта. Такая карта помогает установить задачи класса и выявить его окружение(классы-сотрудники). Для каждого класса создается отдельная карта.

В каждой CRC-карте указывается имя класса, его обязанности (операции) и его сотрудничества (другие классы, в которые он посылает сообщения и от которых он зависит при выполнении своих обязанностей). Сотрудничества подразумевают наличие ассоциаций и зависимостей между классами. Они фиксируются в других моделях – диаграмме последовательности объектов и диаграмме классов.

CRC -карта намеренно сделана простой, даже ее ограниченный размер имеет определенный смысл: если список обязанностей и классов-сотрудников не помещается на карте, то наверное , данный класс надо разделить на несколько классов.

Для оценки модели (диаграммы) классов на основе CRC -карт рекомендуются следующие шаги.

1). Выполняется перекрестный просмотр CRC-карты и диаграммы последовательности объектов. Цель- проверить наличие классов-сотрудников и согласованность информации в обеих моделях.

2). Исследуются обязанности CRC -карты. Цель- определить, предусмотрена ли в карте класса-сотрудника обязанность, которая делегируется ему из данной карты. Например, для -карты «Банкомат», обязанностями являются : читать_карту_клиента(), идентификация_клиента(), проверка_счета(), выдача_денег(), выдача_квитанции(), захват_карты(). Для этой карты выясняем, выполняется ли обязанность (операция) читать_карту_клиента(), которая требует использования класса-сотрудника «Карта-клиента». Это означает, что класс Карта-клиента должен иметь операцию, которая позволяет ему быть прочитанным. Исходя из обязанностей класса «Банкомат» можно судить о множестве классов-сотрудников, имеющих ассоциацию с этим классом: Карта_клиента, База_данных_клиентов, База_данных_счетов, Блок_денег, Блок_квитанций, Блок_карт.

3). Организуется проход по каждой ассоциации (соединению) CRC - карты. Проверяется корректность запросов, выполняемых через соединения. Такая проверка гарантирует, что каждый класс –сотрудник, предоставляющий услугу, получает обновленный запрос. Например, если произошла ошибка и класс «База_данных_клиентов» получает от класса «Банкомат» запрос на проверка_счета(), он не сможет его выполнить – ведь класс «База_данных_клиентов» не знает состояния их счетов.

4).Определяется, требуются ли другие классы, или правильно ли распределены обязанности по классам. Для этого используют проходы по соединениям, исследованные на шаге 3.

5).Определяется, нужно ли объединять часто запрашиваемые обязанности. Например, в любой ситуации используют пару обязанностей – читать_карту_клиента и идентификация_клиента. Их можно объединить в новую обязанность проверка_клиента, которая подразумевает как чтение его карты, так и идентификацию клиента.

6). Шаги 1-5 применяются итеративно, к каждому классу и на каждом шаге эволюции объектно-ориентированной модели.

Тестирование, основанное на сценариях

Тестирование, основанное на сценариях выявляет такой важный тип ошибок, как некорректные спецификации артефактов.

Ошибки из-за неправильной спецификации моделей требований и других моделей анализа и проектирования, означают, что система не выполняет то, чего хочет пользователь.

Тестирование, основанное на сценариях, ориентировано на действия пользователя, а не на действия системы. Это означает фиксацию задач, которые выполняет пользователь, а затем применение их в качестве тестовых вариантов. Задачи пользователя фиксируются с помощью вариантов использования.

Сценарии выполнения событий основного и альтернативного потоков поведения варианта использования обнаруживают ошибки взаимодействия, каждая из которых может быть следствием многих причин.

Рассмотрим, например, проектирование тестов, основанных на сценариях, для текстового редактора.

Рабочие сценарии опишем в виде спецификации варианта использования «*Исправлять черновик*».

Предусловия: обычно печатают черновик, читают его и обнаруживают ошибки, которые не видны на экране. Данный вариант использования описывает события, которые при этом происходят.

- Печатать весь документ.
- Прочитать документ, изменить определенные страницы.
- После внесения изменения страница перепечатывается.
- Иногда перепечатываются несколько страниц.

Этот сценарий определяет как требования тестов, так и требования пользователя.

Требования пользователя очевидны, ему нужны:

- метод для печати отдельной страницы;
- метод для печати диапазона страниц.

В ходе тестирования проверяется редакция текста как до печати, так и после печати. Функция печати может вызвать ошибки в функции редактирования, что означает зависимость этих функций друг от друга.

Вариант использования « *Печатать новую копию*».

Предусловие: кто-то просит пользователя напечатать копию документа.

Предполагаемый сценарий:

- Открыть документ.
- Напечатать документ.
- Закрыть документ.

И в этом случае подход к тестированию почти очевиден, за исключением того, что не определено, откуда появился документ, он был создан в ранней задаче. Означает ли это, что только эта задача влияет на сценарий?

Во многих современных редакторах запоминаются данные о последней печати документа, по умолчанию, эту печать можно повторить. После сценария «Исправлять черновик», достаточно выбрать в меню *Печать*, а затем нажать кнопку *Печать* в диалоговом окне, – в результате повторяется печать последней исправленной страницы. Таким образом, откорректированный сценарий варианта использования «Печатать новую копию» примет следующий вид:

- Открыть документ.
- Выбрать в меню пункт Печать.
- Проверить, что печаталось, и если печатался диапазон страниц, то выбрать опцию *Печатать целый документ*
- Нажать кнопку Печать.
- Закрыть документ.

Этот сценарий указывает возможную ошибку спецификации : редактор не делает того, что пользователь ожидает от него.

Таким образом, при тестировании, основанном на сценариях, разрабатывается набор тестов, покрывающих заданные сценарии. С учетом степени неопределенности, заложенной в сценарии, каждый тест может покрывать один сценарий, несколько сценариев, или, наоборот, часть сценария.

Использование сценариев не требует наличия полной формальной спецификации требований, но зато может потребовать больше времени на разработку и анализ.

Еще одна особенность тестирования сценариев заключается в том, что этот метод направляет тестирование на проверку конкретных режимов использования системы, что позволяет находить дефекты, которые другие методы тестирования по требованиям могут пропустить.

Таким образом, для оценки полноты, правильности и согласованности объектной модели системы, рекомендуется использовать CRC –карты, где можно промоделировать взаимодействие классов, при реализации основных сценариев работы системы.

Основная литература: 1,2,3,4,7.

Контрольные вопросы:

1. Что такое CRC -карта? Как ее применить для тестирования визуальной объектной модели?

2. Поясните особенности тестирования объектно-ориентированных систем и их элементов.

3. В чем заключаются особенности объектно-ориентированного тестирования правильности?

4. Поясните содержание тестирования, основанного на сценариях.

5. Перечислите известные вам методы тестирования взаимодействия классов. Поясните их содержание.

Лабораторная работа №7. Построение диаграммы пригодности, активности, классов, размещения и состояний. Планирование работ

Цель работы – обучение принципам составления плана проектных работ . Как формировать список задач проекта, определять их длительность и зависимости между ними, как вводить в проект информацию об ограничениях по срокам исполнения задач.

Планирование проекта заключается в том, чтобы достаточно точно оценить сроки исполнения и стоимость этих работ. Чем точнее оценка, тем выше качество плана проекта.

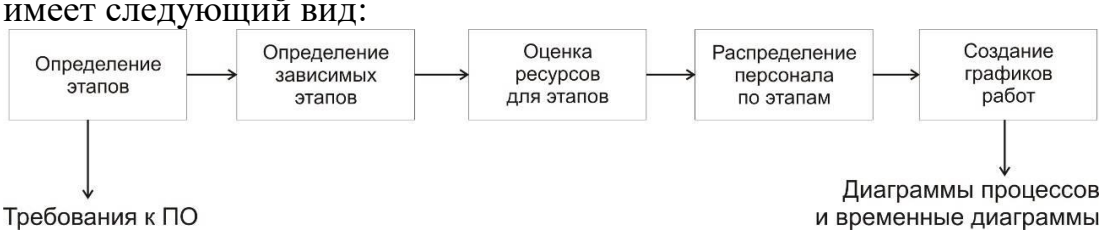
План проекта должен четко показать ресурсы, необходимые для реализации проекта, разделение работ на этапы и временной график выполнения этих этапов. Можно представить план проекта как описание технологического процесса создания ПО. В таком плане обязательно присутствуют ссылки на планы других, разрабатываемых отдельно, видов.

Большинство планов содержат следующие разделы:

1. Введение
2. Организация выполнения проекта.
3. Анализ рисков.
4. Аппаратные и программные ресурсы, необходимые для реализации проекта.
5. Разбиение работ на этапы.
6. График работ.
7. Механизмы мониторинга и контроля за ходом выполнения проекта.

График работ и сетевые диаграммы

Составление графика работ - одна из самых ответственных работ, выполняемых менеджером проекта. Здесь менеджер оценивает длительность проекта, определяет ресурсы для реализации этапов и представляет эти этапы в виде согласованной последовательности. Процесс составления графика работ имеет следующий вид:



Создание графиков работ реализуется в виде временных и сетевых диаграмм, показывающие зависимости между различными этапами проекта.

Разновидностями графиков работ могут быть: этапы проекта, сетевая диаграмма этапов, временная диаграмма деятельности этапов и распределение исполнителей по этапам.

Управление рисками

Важной частью работы менеджера проекта является оценка рисков, которые могут повлиять на график работ или на качество программного продукта, и разработка мероприятий по предотвращению риска.

Выделяются три типа рисков.

1. *Риски для проекта*, которые влияют на график работ или ресурсы.

2. *Риски для разрабатываемого продукта*, влияющие на качество или производительность разрабатываемого программного продукта.

3. *Бизнес-риски*, относящиеся к организации- разработчику или поставщикам.

Процесс управления рисками состоит из четырех стадий.

1. Отделение рисков.

2. Анализ рисков.

3. Планирование рисков.

4.Мониторинг рисков.

Методика выполнения лабораторной работы

Чтобы дать точную оценку, нужно хорошо представлять состав работ проекта, то есть знать, какие именно работы нужно выполнить для получения результата. Только после того, как *составлен список проектных работ*, *оценивается длительность каждой* из них и *выделяются ресурсы*, необходимые для их выполнения. И лишь затем можно *оценить стоимость и сроки исполнения* каждой задачи и, в результате сложения, общую стоимость и срок исполнения всего проекта. Вот почему определение состава работ является первым шагом при планировании проекта.

Определение состава проектных работ начинается с *определения этапов* (или фаз) проекта. Например, в проекте издания журнала могут быть выделены фазы планирования номера, подготовки материалов, верстки и предпечатной подготовки.

После того как состав фаз и их результаты определены, нужно *определить последовательность этих фаз* относительно друг друга и *крайние сроки их исполнения*. Затем нужно выяснить, *из каких работ состоят фазы*, в какой последовательности исполняются эти работы и в какие крайние сроки нужно уложиться при их исполнении. То есть принципы планирования задач внутри фаз повторяют принципы планирования фаз внутри проекта.

Определять состав работ удобно поэтапно. Сначала создается скелет плана работ, состоящий из фаз, их результатов и нескольких основных задач. Потом в план добавляются остальные задачи, определяются их длительности и связи. Затем определяются ключевые даты проекта, которые включают крайние сроки достижения результатов проекта и некоторые другие ограничения по времени. Наконец, в план добавляется дополнительная информация о задачах.

Управление проектом

Методика проведения лабораторной работы связана с типовым процессом управления проектом, и состоит из следующих действий, объединяемых общим названием - *планирование проекта*:

1. Написание предложений по созданию ПО.

Содержат описание целей проекта и способов их достижения. Они также включают в себя оценки финансовых и временных затрат на выполнение проекта.

2. Планирование и составление графика работ по созданию ПО.

На этапе планирования проекта определяются процессы, этапы и получаемые на каждом из этапов результаты, которые должны привести к выполнению проекта.

3. Начальное оценивание стоимости проекта.

Определение начальной стоимости проекта напрямую связано с его планированием, поскольку здесь оцениваются ресурсы, требующиеся для выполнения плана.

4. Контроль за ходом выполнения работ.

Мониторинг проекта- это непрерывный процесс, когда менеджер проекта постоянно отслеживает ход реализации проекта, сравнивая фактические и плановые показатели выполнения работ с их стоимостью.

5. Подбор персонала

Менеджеры проекта обычно обязаны сами подбирать исполнителей для своих проектов, в соответствии с их профессиональным уровнем.

Такой подбор имеет определенные ограничения и не является свободным.

6. Написание отчетов и предложений.

Заключается в написании отчетов о ходе выполнения проекта. В этих отчётах должна быть та информация, которая позволяет четко оценить степень готовности создаваемого ПО.

Эффективное управление программным проектом напрямую зависит не только от плана проекта, но и от таких планов, как: *план качества, план аттестации, план управления конфигурацией, план сопровождения ПО и, наконец, плана по управлению персоналом*.

Процесс планирования проекта можно сформулировать с помощью следующего псевдокода:

Определение проектных ограничений.
Первоначальная оценка параметров проекта.
Определение этапов выполнения проекта и контрольных отметок.
ЦИКЛ (*пока проект не завершится или не будет остановлен*)
Составление графика работ.
Начало выполнения работ.
Ожидание окончания очередного этапа работ.
Отслеживание хода выполнения работ.
Пересмотр оценок параметров проекта.
Изменение графика работ.

Пересмотр проектных ограничений.

ЕСЛИ (*возникла проблема*)

ТО *Пересмотр технических или организационных параметров проекта.*

ВСЕ-ЕСЛИ

ВСЕ-ЦИКЛ

Основная литература – 1,2,3,4,5,7.

Контрольные вопросы

1**.** Что такое управление проектом? Составляющие управления проектом.

2. Объясните, почему нематериальность програмных систем порождает особые проблемы в процессе управления программным проектом?

3. Как определять зависимости между задачами.

4. Какие типы зависимостей между задачами существуют и как они влияют на расчет календарного плана проекта программой.

5. Объясните суть и назначение графических способов представления графиков планов работ.

6. Как оптимизировать план работ проекта?

7. Как анализировать распределение затрат по фазам проекта, типам работ, типам трудозатрат и типам ресурсов?

8. Что такое управление проектом?Составляющие управления проектом.

9. Объясните, почему нематериальность програмных систем порождает особые проблемы в процессе управления программным проектом?

10. Объясните, почему процесс планирования проекта является операционным и почему план должен постоянно пересматриваться в течение всего срока выполнения проекта?

11. Определите, в дополнение к приведенным рискам, еще на ваш взгляд заслуживающих внимания.

12. Объясните суть и назначение графических способов представления графиков планов работ.

Список литературы

Основная:

1 Орлов С.А., Технология разработки программного обеспечения. Учебник. - СПб: Питер, 2002,2012.

2 С.В. Маклаков BPWin, и ERWin. CASE-разработки информационных систем. - М.:ДИАЛОГ-МИФИ, 2000 - 256 с.

3 Грейди Буч, Джеймс Рамбо, Айвар Джекобсон, Язык UML. Руководство пользователя: Пер. с англ - М.: ДМК Пресс, 2001.

4 Марка Д.А., Мак Гоуэн К. Методология структурного анализа и проектирования. М., "МетаТехнология", 1993.

5 Калянов Г.Н. CASE. Структурный системный анализ (автоматизация и применение). М., "Лори", 1996.

6 Шлеер С., Меллор С. Объектно-ориентированный анализ: моделирование мира в состояниях. Киев, "Диалектика", 1993.

7 Гагарина, Л.Г. Технология разработки программного обеспечения.-М.: ФОРУМ-ИНФРА-М, 2009,2011,2012.

8 Панюкова Т.А. Проектирование программных средств.-М.: «ЛИБРОКОМ»,2012

9 Крылов Е.В. Техника разработки программ. Кн.2. Технология, надежность и качество программного обеспечения. - М.,2008

10 Скопин И.Н. Основы менеджмента программных проектов. -М.: «Бином», 2004,2009,2012

11 Кулямин В.В. Технологии программирования. Компонентный подход.-М.,2007,2014

Дополнительная:

12 Шилдт Г. JAVA Полное руководство. -М.: «Вильямс», 2012,2013

13 Лафоре Р. Структуры данных и алгоритмы JAVA. - СПб.: «Питер», 2011

14 Гергель В.П. Теория и практика параллельных вычислений. - М.: «Интернет-УИТ: Бином», 2007,2013

15 Черников Б.В. Управление качеством программного обеспечения. - М.: «Форум», «Инфра-М», 2012