МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Пятигорский институт (филиал)

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ВЫПОЛНЕНИЮ ЛАБОРАТОРНЫХ РАБОТ ПО ДИСЦИПЛИНЕ СИСТЕМНАЯ ИНЖЕНЕРИЯ

Направление подготовки	**09.04.02 Информационные системы и технологии**
Направленность (профиль)	**Технологии работы с данными и знаниями, анализ информации**
Квалификация выпускника	Магистр

Пятигорск, 2025

СОДЕРЖАНИЕ

Введение....3

1. Цель и задачи изучения дисциплины....5

2. Оборудование и материалы....5

3. Указания по технике безопасности....5

4. НАИМЕНОВАНИЕ ЛАБОРАТОРНЫХ РАБОТ....6

5. СОДЕРЖАНИЕ ЛАБОРАТОРНЫХ РАБОТ....6

Лабораторная работа 1. Применение принципов системной инженерии в проектировании информационных систем. Разработка технического задания на проектирование информационной системы....6

Лабораторная работа 2. Применение принципов системной инженерии в проектировании информационных систем. Средства структурного анализа информационных систем. 17

Лабораторная работа 3. Системный инжиниринг проекта информационной системы 23

Лабораторная работа 4. Инжиниринг и реинжиниринг архитектуры информационных систем. Спецификация программного обеспечения при объектном подходе. 27

6. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ 29

6.1. Перечень основной и дополнительной литературы, необходимой для освоения дисциплины 29

6.2. Перечень учебно-методического обеспечения самостоятельной работы обучающихся по дисциплине 30

6.3. Перечень ресурсов информационно-телекоммуникационной сети Интернет, необходимых для освоения дисциплины 30

ВВЕДЕНИЕ

В методических указаниях содержатся материалы, необходимые для самостоятельной подготовки студентов к выполнению лабораторных работ. В описание лабораторных работ включены цель работы, порядок ее выполнения, рассмотрены теоретические вопросы, связанные с реализацией поставленных задач, приведена необходимая литература.

Методические указания посвящены курсу «Системная инженерия». Самое современное определение системной инженерии дано в Guide to the Systems Engineering Body of Knowledge (руководство по корпусу знаний системной инженерии). Короткое определение понятия системная инженерия: это междисциплинарный подход и способы обеспечения воплощения успешной системы (Systems engineering is an interdisciplinary approach and means to enable the realization of successful systems). В этом определении можно подчеркнуть следующие моменты:

1. Успешные системы — это то, чем занимается системная инженерия. Слово “успешные” тут крайне важно и означает, что система должна удовлетворить нужды заказчиков, пользователей и других стейкхолдеров (стейкхолдеры — это те, кто затрагивается системой, или кто затрагивает систему). Успех —э то когда системой все довольны.

2. Слово “системы” используется в очень специальном значении: это «системы» из системного подхода. Если вы сказали «система» про компьютер, то это автоматически влечёт за собой разговор про стейкхолдеров и их интересы, требования и архитектуру, жизненный цикл и т.д.

3. Междисциплинарный подход — системная инженерия претендует на то, что она работает со всеми остальными инженерными специальностями (впрочем, не только инженерными). «Подход» обычно означает какие-то наработки в одной предметной области, которые можно перенести на другие предметные области. Междисциплинарность — это очень сильное заявление, оно означает, что системная инженерия может в одну упряжку впрячь коня и трепетную лань (например, инженеров-механиков, баллистиков, криогенщиков, психологов, медиков, астрономов, программистов и т.д. в проектах пилотируемой космонавтики).

4. Слово “воплощение” (realization, “перевод в реальность”) означает буквально это: создание материальной (из вещества и полей) успешной системы.

По-английски “системная инженерия” — systems engineering, хотя более ранние написания были как system engineering. Правильная интерпретация (и правильный перевод) — именно «системная» (подразумевающая использование системного подхода) инженерия, а не «инженерия систем» (engineering of systems) — когда любой «объект» называется «системой», но не используется системный подход во всей его полноте.

Из системной инженерии квалификатор "системный" без изменения смысла понятия убрать нельзя — системная инженерия это инженерия с системным мышлением.

Более длинное определение включает ещё одну фразу: «Она фокусируется на целостном и одновременном/параллельном понимании нужд стейкхолдеров; исследовании возможностей; документировании требований; и синтезировании, проверке, приёмке и постепенном появлении инженерных решений, в то время как в расчёт принимается полная проблема, от исследования концепции системы до вывода системы из эксплуатации» (It focuses on holistically and concurrently understanding stakeholder needs; exploring opportunities; documenting requirements; and synthesizing, verifying, validating, and evolving solutions while considering the complete problem, from system concept exploration through system disposal). Тут нужно подчеркнуть:

1 Целокупность, которая подчёркнута многократно — от “междисциплинарности” в первой половине определения до целостности всех действий по созданию системы

во второй половине определения, до целостности/полноты проблемы, до охвата всего жизненного цикла системы “от рождения до смерти”. Целостность (полнота охвата всех частей целевой системы согласованным их целым), междисциплинарность (полнота охвата всех дисциплин) — это ключевое, что отличает системную инженерию от всех остальных инженерных дисциплин.

2. Параллельность выполнения самых разных практик (а не последовательное выполнение их во времени, как можно было бы подумать, прочитав перечисление практик)

3. Много особенностей (различение нужд пользователей и требований, проверки и приёмки, упор на синтез для противопоставления “аналитическим” дисциплинам и т.д.).

Ответственность за всю систему как целое (whole system) и связанная с этим межпредметность/междисциплинарность (interdisciplinary) подхода к другим инженериям (механической, электрической, программной, предприятия и т.д.) отличают системную инженерию от всех других инженерных дисциплин.

Представим себе ледовую буровую платформу: Сотни тысяч тонн металла, бетона, пластмассы, необходимых расходных материалов, обученная вахта должны собраться вместе далеко в море среди льда и в строго определённый момент эта огромная конструкция должна начать согласованно работать — и не просто работать, а приносить прибыль и обеспечивать безопасность в части загрязнения окружающей среды и здоровья находящейся на платформе вахты. Какая инженерная дисциплина должна учесть результаты работ всех других инженерных дисциплин — собрать в единое целое данные ледовой обстановки, санитарных норм в помещениях для обслуживающего персонала, обеспечение электричеством попавших туда компьютерных серверов, характеристики эти серверов и программное обеспечение? Кто озаботится учётом в конструкции платформы изменений в длине металлоконструкций за счёт разницы суточных температур и одновременно установкой акустических датчиков на трубах, которые прослушивают шорох песка, чтобы по этому шороху можно было определить износ труб?

Системная инженерия как раз и является той дисциплиной, которая ответственна за обеспечение целостности в инженерном проекте: именно системные инженеры проектируют нефтяную платформу как успешное (безопасное, надёжное, прибыльное, ремонтопригодное и т.д. — разным людям нужно от этой платформы разное) целое, потом раздают части работы инженерам по специальностям (инженерам-строителям, машиностроителям, инженерам-электрикам, компьютерщикам/айтишникам и т.д.), а затем собирают результаты их работ так, чтобы получить работоспособную и надёжную систему.

Это главный признак, отличающий системных инженеров от всех других инженеров: они отвечают за проект в целом, сразу во всех его деталях как в части деталей-частей, так и в части использования детальных знаний отдельных дисциплин. Они ответственны за то, чтобы не было пропущено какой-нибудь мелочи, ведущей к провалу.

Самолёт — это много-много кусков металла и пластика, синхронно летящих на скорости 900км/час (0.85 от скорости звука, это типовая скорость Boeing 787 Dreamliner) на высоте 10км. Системный инженер — это тот, кто придумал, как обеспечить их надёжный и экономичный совместный полёт, увязав самые разные требования (грузоподъёмность, расход топлива, дальность полёта, шум при взлёте и посадке, требования к длине разбега и посадки, необходимость лёгкого обслуживания на земле, отсутствие обледенения, безопасность людей на борту, и т.д. и т.п.), при этом требования выдвигались самыми разными людьми, представляющими самые разные профессиональные и общественные группы. Пара-тройка миллионов деталей

изготавливается и собирается в одно изделие — и самолёт летит, обеспечивая комфорт пассажирам и прибыль владельцам.

Системная инженерия решает задачу преодоления инженерной сложности (которая определяется главным образом как число различных элементов, которые должны провзаимодействовать друг с другом, чтобы получить целевую систему — и сложные системы не помещаются ни в какую самую умную голову, требуется специальная дисциплина, чтобы собирать результаты деятельности самых разных инженеров в одно работоспособное целое.

1. ЦЕЛЬ И ЗАДАЧИ ИЗУЧЕНИЯ ДИСЦИПЛИНЫ

Целью освоения дисциплины «Системная инженерия» является формирование набора профессиональных компетенций будущего магистра по направлению подготовки 09.04.02 «Информационные системы и технологии» для решения прикладных задач в рамках магистерской программы «Технологии работы с данными и знаниями, анализ информации».

Задачи освоения дисциплины: изучение основных понятий системной инженерии, освоение методов и инструментов системной инженерии.

2. ОБОРУДОВАНИЕ И МАТЕРИАЛЫ

Аппаратные средства: персональный компьютер;

Программные средства: ОС MS Windows; MS Visual Studio, MS Office.

Учебный класс оснащен IBM-совместимыми компьютерами, объединенными в локальную сеть. Локальная сеть учебного класса имеет постоянный доступ к сети Internet по выделенной линии. Для проведения лабораторных работ необходимо следующее программное обеспечение: операционная система MS Windows, пакет офисных программ MS Office, пакет MS Visual Studio.

3. УКАЗАНИЯ ПО ТЕХНИКЕ БЕЗОПАСНОСТИ

Перед началом работы следует убедиться в исправности электропроводки, выключателей, штепсельных розеток, при помощи которых оборудование включается в сеть, наличии заземления компьютера, его работоспособности.

Для снижения или предотвращения влияния опасных и вредных факторов необходимо соблюдать санитарные правила и нормы, гигиенические требования к персональным электронно-вычислительным машинам.

Во избежание повреждения изоляции проводов и возникновения коротких замыканий не разрешается: вешать что-либо на провода, закрашивать и белить шнуры и провода, закладывать провода и шнуры за газовые и водопроводные трубы, за батареи отопительной системы, выдергивать штепсельную вилку из розетки за шнур, усилие должно быть приложено к корпусу вилки.

Для исключения поражения электрическим током запрещается: часто включать и выключать компьютер без необходимости, прикасаться к экрану и к тыльной стороне блоков компьютера, работать на средствах вычислительной техники и периферийном оборудовании мокрыми руками, работать на средствах вычислительной техники и периферийном оборудовании, имеющих нарушения целостности корпуса, нарушения изоляции проводов, неисправную индикацию включения питания, с признаками электрического напряжения на корпусе, класть на средства вычислительной техники и периферийном оборудовании посторонние предметы.

Запрещается под напряжением очищать от пыли и загрязнения электрооборудование.

Во избежание поражения электрическим током, при пользовании электроприборами нельзя касаться одновременно каких-либо трубопроводов, батарей отопления, металлических конструкций, соединенных с землей.

После окончания работы необходимо обесточить все средства вычислительной техники и периферийное оборудование. В случае непрерывного учебного процесса необходимо оставить включенными только необходимое оборудование.

4. НАИМЕНОВАНИЕ ЛАБОРАТОРНЫХ РАБОТ

Лабораторная работа 1. Применение принципов системной инженерии в проектировании информационных систем. Разработка технического задания на проектирование информационной системы

Лабораторная работа 2. Применение принципов системной инженерии в проектировании информационных систем. Средства структурного анализа информационных систем.

Лабораторная работа 3. Системный инжиниринг проекта информационной системы

Лабораторная работа 4. Инжиниринг и реинжиниринг архитектуры информационных систем. Спецификация программного обеспечения при объектном подходе.

5. СОДЕРЖАНИЕ ЛАБОРАТОРНЫХ РАБОТ

Лабораторная работа 1. Применение принципов системной инженерии в проектировании информационных систем. Разработка технического задания на проектирование информационной системы

Цель работы: составить и проанализировать требования к информационной системе; разработать техническое задание на проектирование информационной системы.

Основы теории

Проблемы, которые приходится решать специалистам в процессе создания программного обеспечения, очень сложны. Природа этих проблем не всегда ясна, особенно если разрабатываемая программная система инновационная. В частности, трудно чётко описать те действия, которые должна выполнять система. Описание функциональных возможностей и ограничений, накладываемых на систему, называется требованиями к этой системе, а сам процесс формирования, анализа, документирования и проверки этих функциональных возможностей и ограничений – разработкой требований.

Требования подразделяются на пользовательские и системные. Пользовательские требования – это описание на естественном языке (плюс поясняющие диаграммы) функций, выполняемых системой, и ограничений, накладываемых на неё. Системные требования – это описание особенностей системы (архитектура системы, требования к параметрам оборудования и т.д.), необходимых для эффективной реализации требований пользователя.

Разработка требований

Разработка требований — это процесс, включающий мероприятия, необходимые для создания и утверждения документа, содержащего спецификацию системных требований. Различают четыре основных этапа процесса разработки требований:

- анализ технической осуществимости создания системы,
- формирование и анализ требований,
- специфицирование требований и создание соответствующей документации,
- аттестация этих требований.

На рисунке 1.1 показаны взаимосвязи между этими этапами и результаты, сопровождающие каждый этап процесса разработки системных требований.

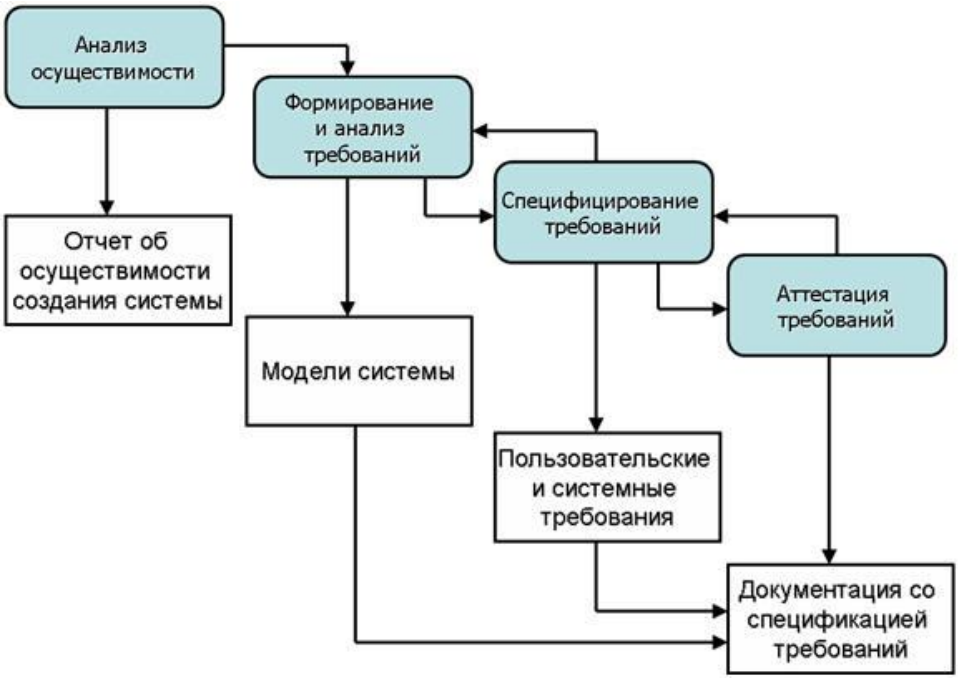

Рисунок 1.1 - Процесс разработки требований

Но поскольку в процессе разработки системы в силу разнообразных причин требования могут меняться, управление требованиями, т.е. процесс управления изменениями системных требований, является необходимой составной частью деятельности по их разработке.

Формирование и анализ требований

Следующим этапом процесса разработки требований является формирование (определение) и анализ требований.

Обобщенная модель процесса формирования и анализа требований показана на рисунке 1.2. Каждая организация использует собственный вариант этой модели, зависящий от “местных факторов”: опыта работы коллектива разработчиков, типа разрабатываемой системы, используемых стандартов и т.д.

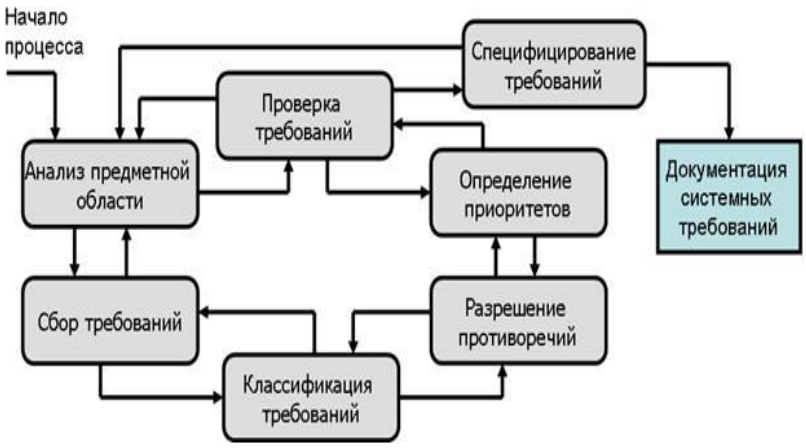

Рисунок 1.2 - Процесс формирования и анализа требований

Процесс формирования и анализа требований проходит через ряд этапов.

1. Анализ предметной области. Аналитики должны изучить предметную область, где будет эксплуатироваться система.

2. Сбор требований. Это процесс взаимодействия с лицами, формирующими требования. Во время этого процесса продолжается анализ предметной области.

3. Классификация требований. На этом этапе бесформенный набор требований преобразуется в логически связанные группы требований.

4. Разрешение противоречий. Без сомнения, требования многочисленных лиц, занятых в процессе формирования требований, будут противоречивыми. На этом этапе определяются и разрешаются противоречия различного рода.

5. Назначение приоритетов. В любом наборе требований одни из них будут более важны, чем другие. На этом этапе совместно с лицами, формирующими требования, определяются наиболее важные требования.

6. Проверка требований. На этом этапе определяется их полнота, последовательность и непротиворечивость.

Процесс формирования и анализа требований циклический, с обратной связью от одного этапа к другому. Цикл начинается с анализа предметной области и заканчивается проверкой требований. Понимание требований предметной области увеличивается в каждом цикле процесса формирования требований.

Рассмотрим три основных подхода к формированию требований: метод, основанный на множестве опорных точек зрения, сценарии и этнографический метод.

Опорные точки зрения

Подход с использованием различных опорных точек зрения к разработке требований признает различные (опорные) точки зрения на проблему и использует их в качестве основы построения и организации как процесса формирования требований, так и непосредственно самих требований.

Различные методы предлагают разные трактовки выражения "точка зрения". Точки зрения можно трактовать следующим образом.

1. Как источник информации о системных данных. В этом случае на основе опорных точек зрения строится модель создания и использования данных в системе. В процессе формирования требований отбираются все такие точки зрения (и на их основе определяются данные), которые будут созданы или использованы при работе системы, а также способы обработки этих данных.

2. Как структура представлений. В этом случае точки зрения рассматриваются как особая часть модели системы. Например, на основе различных точек зрения могут разрабатываться модели "сущность-связь", модели конечного автомата и т.д.

3. Как получатели системных сервисов. В этом случае точки зрения являются внешними (относительно системы) получателями системных сервисов. Точки зрения помогают определить данные, необходимые для выполнения системных сервисов или их управления.

Наиболее эффективным подходом к анализу таких систем является использование внешних опорных точек зрения. На основе этого подхода разработан метод VORD (Viewpoint-Oriented Requirements Definition — определение требований на основе точек зрения) для формирования и анализа требований. Основные этапы метода VORD показаны на рисунке 1.3.

1. Идентификация точек зрения, получающих системные сервисы, и идентификация сервисов, соответствующих каждой точке зрения.

2. Структурирование точек зрения — создание иерархии сгруппированных точек зрения. Общесистемные сервисы предоставляются более высоким уровням иерархии и наследуются точками зрения низшего уровня.

3. Документирование опорных точек зрения, которое заключается в точном описании идентифицированных точек зрения и сервисов.

4. Отображение системы точек зрения, которая показывает системные объекты, определенные на основе информации, заключенной в опорных точках зрения.

Рисунок 1.3 - Метод VORD

Пример. Рассмотрим использование метода VORD на первых трех шагах анализа требований для системы системы поддержки заказа и учета товаров в бакалейной лавке. В бакалейной лавке для каждого товара фиксируется место хранения (определенная полка), количество товара и его поставщик. Система поддержки заказа и учета товаров должна обеспечивать добавление информации о новом товаре, изменение или удаление информации об имеющемся товаре, хранение (добавление, изменение и удаление) информации о поставщиках, включающей в себя название фирмы, ее адрес и телефон. При помощи системы составляются заказы поставщикам. Каждый заказ может содержать несколько позиций, в каждой позиции указываются наименование товара и его количество в заказе. Система по требованию пользователя формирует и выдает на печать следующую справочную информацию:

- список всех товаров;
- список товаров, имеющихся в наличии;
- список товаров, количество которых необходимо пополнить;
- список товаров, поставляемых данным поставщиком.

Первым шагом в формировании требований является идентификация опорных точек зрения. Во всех методах формирования требований, основанных на использовании точек зрения, начальная идентификация является наиболее трудной задачей. Один из подходов к идентификации точек зрения — метод "мозговой атаки", когда определяются потенциальные системные сервисы и организации, взаимодействующие с системой. Организуется встреча лиц, участвующих в формировании требований, которые предлагают свои точки зрения. Эти точки зрения представляются в виде диаграммы, состоящей из ряда круговых областей, отображающих возможные точки зрения (рис. 4). Во время "мозговой атаки" необходимо идентифицировать потенциальные опорные точки зрения, системные сервисы, входные данные, нефункциональные требования, управляющие события и исключительные ситуации.

Следующей стадией процесса формирования требований будет идентификация опорных точек зрения (на рисунке 1.4 показаны в виде темных круговых областей) и сервисов (показаны в виде затененных областей). Сервисы должны соответствовать опорным точкам зрения. Но могут быть сервисы, которые не поставлены им в соответствие. Это означает, что на начальном этапе "мозговой атаки" некоторые опорные точки зрения не были идентифицированы.

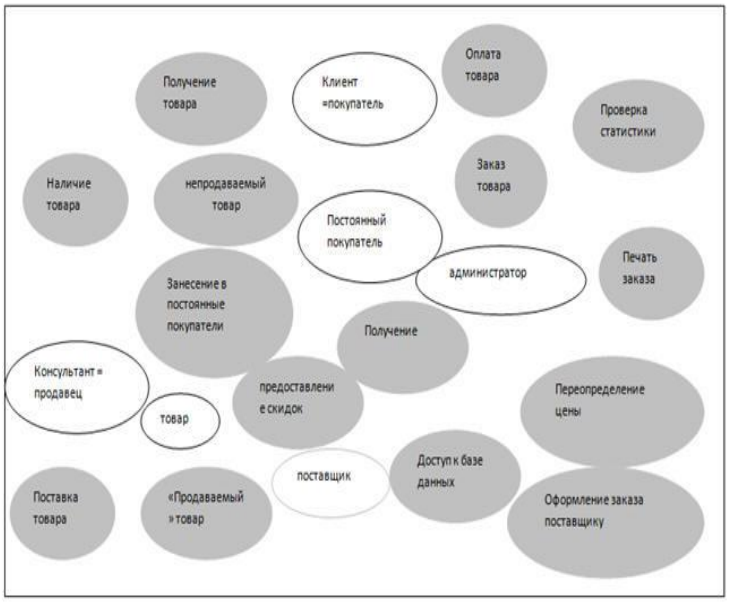

Рисунок 1.4 - Диаграмма идентификации точек зрения

В таблице 1.1 показано распределение сервисов для некоторых идентифицированных на рисунке 1.4 точек зрения. Один и тот же сервис может быть соотнесен с несколькими точками зрения.

Таблица 1.1 - Сервисы, соотнесенные с точками зрения

клиент	**покупатель**	**постоянный покупатель**	**товар**	**поставщик**	**продавец**	**администратор**
Проверка наличия товара	Занесение в список клиентов	Получение скидки	Прием товара	Занесение в базу данных	Продажа товара	Доступ к базе данных
Покупка товара		Получение информации	Занесение в базу данных		Печать чека	Проверка статистики
Получение чека			Назначение цены		Доступ к каталогу	Переопределение цены
Заказ товара			Переопределение цены		Проверка наличия товара	Оформление заказа
Занесение покупателя и суммы покупки в базу данных			«Покупаемый» или «непокупаемый» товар		Оформление заказа покупателю	Печать заказа

Информация, извлеченная из точек зрения, используется для заполнения форм шаблонов точек зрения и организации точек зрения в иерархию наследования. Это позволяет увидеть общие точки зрения и повторно использовать информацию в иерархии наследования. Сервисы, данные и управляющая информация наследуются подмножеством точек

зрения. На рисунке 1.5 показана часть иерархии точек зрения для системы поддержки заказа и учета товаров.

Рисунок 1.5 - Иерархия точек зрения

Аттестация требований

Аттестация должна продемонстрировать, что требования действительно определяют ту систему, которую хочет иметь заказчик. Проверка требований важна, так как ошибки в спецификации требований могут привести к переделке системы и большим затратам, если будут обнаружены во время процесса разработки системы или после введения ее в эксплуатацию. Стоимость внесения в систему изменений, необходимых для устранения ошибок в требованиях, намного выше, чем исправление ошибок проектирования или кодирования. Причина в том, что изменение требований обычно влечет за собой значительные изменения в системе, после внесения которых она должна пройти повторное тестирование.

Во время процесса аттестации должны быть выполнены различные типы проверок требований.

1. Проверка правильности требований. Пользователь может считать, что система необходима для выполнения некоторых определенных функций. Однако дальнейшие размышления и анализ могут привести к необходимости введения дополнительных или новых функций. Системы предназначены для разных пользователей с различными потребностями, и поэтому набор требований будет представлять собой некоторый компромисс между требованиями пользователей системы.

2. Проверка на непротиворечивость. Спецификация требований не должна содержать противоречий. Это означает, что в требованиях не должно быть противоречащих друг другу ограничений или различных описаний одной и той же системной функции.

3. Проверка на полноту. Спецификация требований должна содержать требования, которые определяют все системные функции и ограничения, налагаемые на систему.

4. Проверка на выполнимость. На основе знания существующих технологий требования должны быть проверены на возможность их реального выполнения. Здесь также проверяются возможности финансирования и график разработки системы.

Существует ряд методов аттестации требований, которые можно использовать совместно или каждый в отдельности.

1. Обзор требований. Требования системно анализируются рецензентами.

2. Прототипирование. На этом этапе прототип системы демонстрируется конечным пользователям и заказчику. Они могут экспериментировать с этим прототипом, чтобы убедиться, что он отвечает их потребностям.

3. Генерация тестовых сценариев. В идеале требования должны быть такими, чтобы их реализацию можно было протестировать. Если тесты для требований разрабатываются как часть процесса аттестации, то часто это позволяет обнаружить проблемы в

спецификации. Если такие тесты сложно или невозможно разработать, то обычно это означает, что требования трудно выполнить и поэтому необходимо их пересмотреть.

4. Автоматизированный анализ непротиворечивости. Если требования представлены в виде структурных или формальных системных моделей, можно использовать инструментальные CASE-средства для проверки непротиворечивости моделей. Для автоматизированной проверки непротиворечивости необходимо построить базу данных требований и затем проверить все требования в этой базе данных. Анализатор требований готовит отчет обо всех обнаруженных противоречиях.

Пользовательские и системные требования

На основании полученных моделей строятся пользовательские требования, т.е. как было сказано в начале описание на естественном языке функции, выполняемых системой, и ограничений, накладываемых на неё.

Пользовательские требования должны описывать внешнее поведение системы, основные функции и сервисы предоставляемые системой, её нефункциональные свойства. Необходимо выделить опорные точки зрения и сгруппировать требования в соответствии с ними. Пользовательские требования можно оформить как простым перечислением, так и используя нотацию вариантов использования.

Далее составляются системные требования. Они включают в себя:

1. Требования к архитектуре системы. Например, число и размещение хранилищ и серверов приложений.

2. Требования к параметрам оборудования. Например, частота процессоров серверов и клиентов, объём хранилищ, размер оперативной и видео памяти, пропускная способность канала и т.д.

3. Требования к параметрам системы. Например, время отклика на действие пользователя, максимальный размер передаваемого файла, максимальная скорость передачи данных, максимальное число одновременно работающих пользователей и т.д.

4. Требования к программному интерфейсу.

5. Требования к структуре системы. Например, Масштабируемость, распределённость, модульность, открытость.

- масштабируемость – возможность распространения системы на большое количество машин, не приводящая к потере работоспособности и эффективности, при этом способность системы наращивать свою мощность должна определяться только мощностью соответствующего аппаратного обеспечения.

- распределенность - система должна поддерживать распределённое хранение данных.

- модульность - система должна состоять из отдельных модулей, интегрированных между собой.

- открытость - наличие открытых интерфейсов для возможной доработки и интеграции с другими системами.

6. Требования по взаимодействию и интеграции с другими системами. Например, использование общей базы данных, возможность получения данных из баз данных определённых систем и т.д.

Техническое задание

Техническое задание представляет собой документ, в котором сформулированы основные цели разработки, требования к программному продукту, определены сроки и этапы разработки и регламентирован процесс приемо-сдаточных испытаний. В разработке технического задания участвуют как представители заказчика, так и представители исполнителя. В основе этого документа лежат исходные требования заказчика, анализ передовых достижений техники, результаты выполнения научно-исследовательских работ, предпроектных исследований, научного прогнозирования и т. п.

Порядок разработки технического задания

Разработка технического задания выполняется в следующей последовательности. Прежде всего устанавливают набор выполняемых функций, а также перечень и характеристики исходных данных. Затем определяют перечень результатов, их характеристики и способы представления.

Далее уточняют среду функционирования программного обеспечения: конкретную комплектацию и параметры технических средств, версию используемой операционной системы и, возможно, версии и параметры другого установленного программного обеспечения, с которым предстоит взаимодействовать будущему программному продукту.

В случаях, когда разрабатываемое программное обеспечение собирает и хранит некоторую информацию или включается в управление каким-либо техническим процессом, необходимо также четко регламентировать действия программы в случае сбоев оборудования и энергоснабжения.

1. Общие положения

1.1. Техническое задание оформляют в соответствии со стандартом ГОСТ 19.201-78, ГОСТ 34.602-89.

1.2. Для внесения изменений и дополнений в техническое задние на последующих стадиях разработки программы или программного изделия выпускают дополнение к нему. Согласование и утверждение дополнения к техническому заданию проводят в том же порядке, который установлен для технического задания.

1.3. Техническое задание должно содержать следующие разделы:

1) общие сведения;

2) назначение и цели создания (развития) системы;

3) характеристика объектов автоматизации;

4) требования к системе;

5) состав и содержание работ по созданию системы;

6) порядок контроля и приемки системы;

7) требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие;

8) требования к документированию;

9) источники разработки.

В зависимости от особенностей программы или программного изделия допускается уточнять содержание разделов, вводить новые разделы или объединять отдельные из них. При необходимости допускается в техническое задание включать приложения.

2. Содержание разделов

2.1. В разделе «Общие сведения» указывают:

1) полное наименование системы и ее условное обозначение;

2) шифр темы или шифр (номер) договора;

3) наименование предприятий (объединений) разработчика и заказчика (пользователя) системы и их реквизиты;

4) перечень документов, на основании которых создается система, кем и когда утверждены эти документы;

5) плановые сроки начала и окончания работы по созданию системы;

6) сведения об источниках и порядке финансирования работ;

7) порядок оформления и предъявления заказчику результатов работ по созданию системы (ее частей), по изготовлению и наладке отдельных средств (технических, программных, информационных) и программно-технических (программно-методических) комплексов системы.

2.2. Раздел «Назначение и цели создания (развития) системы» состоит из подразделов:

1) назначение системы;

2) цели создания системы.

2.2.1. В подразделе «Назначение системы» указывают вид автоматизируемой деятельности (управление, проектирование и т. п.) и перечень объектов автоматизации (объектов), на которых предполагается ее использовать.

2.2.2. В подразделе «Цели создания системы» приводят наименования и требуемые значения технических, технологических, производственно-экономических или других показателей объекта автоматизации, которые должны быть достигнуты в результате создания ИС, и указывают критерии оценки достижения целей создания системы.

2.3. В разделе «Характеристики объекта автоматизации» приводят:

1) краткие сведения об объекте автоматизации или ссылки на документы, содержащие такую информацию;

2) сведения об условиях эксплуатации объекта автоматизации и характеристиках окружающей среды.

Примечание: Для САПР в разделе дополнительно приводят основные параметры и характеристики объектов проектирования.

2.4. Раздел «Требования к системе» состоит из следующих подразделов:

1) требования к системе в целом;

2) требования к функциям (задачам), выполняемым системой;

3) требования к видам обеспечения.

Состав требований к системе, включаемых в данный раздел ТЗ на ИС, устанавливают в зависимости от вида, назначения, специфических особенностей и условий функционирования конкретной системы. В каждом подразделе приводят ссылки на действующие НТД, определяющие требования к системам соответствующего вида.

2.3.1. В подразделе «Требования к системе в целом» указывают:

- требования к структуре и функционированию системы;
- требования к численности и квалификации персонала системы и режиму его работы;
- показатели назначения;
- требования к надежности;
- требования безопасности;
- требования к эргономике и технической эстетике;
- требования к транспортабельности для подвижных АС;
- требования к эксплуатации, техническому обслуживанию, ремонту и хранению компонентов системы;
- требования к защите информации от несанкционированного доступа;
- требования по сохранности информации при авариях;
- требования к защите от влияния внешних воздействий;
- требования к патентной чистоте;
- требования по стандартизации и унификации;
- дополнительные требования.

2.3.1.1. В требованиях к структуре и функционированию системы приводят:

1) перечень подсистем, их назначение и основные характеристики, требования к числу уровней иерархии и степени централизации системы;

2) требования к способам и средствам связи для информационного обмена между компонентами системы;

3) требования к характеристикам взаимосвязей создаваемой системы со смежными системами, требования к ее совместимости, в том числе указания о способах обмена информацией (автоматически, пересылкой документов, по телефону и т. п.);

4) требования к режимам функционирования системы;

5) требования по диагностированию системы;

6) перспективы развития, модернизации системы.

2.3.1.2. В требованиях к показателям назначения ИС приводят значения параметров, характеризующие степень соответствия системы ее назначению.

Для ИС управления указывают:

- допустимые пределы модернизации и развития системы;
- временные характеристики, при которых сохраняется назначение системы.

2.3.2. В подразделе «Требования к видам обеспечения» в зависимости от вида системы приводят требования к математическому, информационному, лингвистическому, программному, техническому, метрологическому, организационному, методическому и другие видам обеспечения системы.

2.3.2.1. Для математического обеспечения системы приводят требования к составу, области применения (ограничения) и способам, использования в системе математических методов и моделей, типовых алгоритмов и алгоритмов, подлежащих разработке.

2.3.2.2. Для информационного обеспечения системы приводят требования:

1) к составу, структуре и способам организации данных в системе;

2) к информационному обмену между компонентами системы;

3) к информационной совместимости со смежными системами;

4) по использованию общесоюзных и зарегистрированных республиканских, отраслевых классификаторов, унифицированных документов и классификаторов, действующих на данном предприятии;

5) по применению систем управления базами данных;

6) к структуре процесса сбора, обработки, передачи данных в системе и представлению данных;

7) к защите данных от разрушений при авариях и сбоях в электропитании системы;

8) к контролю, хранению, обновлению и восстановлению данных;

9) к процедуре придания юридической силы документам, продуцируемым техническими средствами ИС (в соответствии с ГОСТ 6.10.4).

2.3.2.3. Для лингвистического обеспечения системы приводят требования к применению в системе языков программирования высокого уровня, языков взаимодействия пользователей и технических средств системы, а также требования к кодированию и декодированию данных, к языкам ввода-вывода данных, языкам манипулирования данными, средствам описания предметной области (объекта автоматизации), к способам организации диалога.

2.3.2.4. Для программного обеспечения системы приводят перечень покупных программных средств, а также требования:

1) к независимости программных средств от используемых СВТ и операционной среды;

2) к качеству программных средств, а также к способам его обеспечения и контроля;

3) по необходимости согласования вновь разрабатываемых программных средств с фондом алгоритмов и программ.

2.3.2.5. Для технического обеспечения системы приводят требования:

1) к видам технических средств, в том числе к видам комплексов технических средств, программно-технических комплексов и других комплектующих изделий, допустимых к использованию в системе;

2) к функциональным, конструктивным и эксплуатационным характеристикам средств технического обеспечения системы.

2.3.2.6. Для организационного обеспечения приводят требования к структуре и функциям подразделений, участвующих в функционировании системы или обеспечивающих эксплуатацию;•

2.3.2.7. Для методического обеспечения ИС приводят требования к составу нормативно-технической документации системы (перечень применяемых при ее функционировании стандартов, нормативов, методик и т. п.).

2.4. Раздел «Состав и содержание работ по созданию (развитию) системы» должен содержать перечень стадий и этапов работ по созданию системы в соответствии с ГОСТ 24.601, сроки их выполнения, перечень организаций - исполнителей работ, ссылки на

документы, подтверждающие согласие этих организаций на участие в создании системы, или запись, определяющую ответственного (заказчик или разработчик) за проведение этих работ.

2.5. В разделе «Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие» необходимо привести перечень основных мероприятий и их исполнителей, которые следует выполнить при подготовке объекта автоматизации к вводу ИС в действие.

2.6. В состав ТЗ на проектирование ИС при наличии утвержденных методик включают приложения, содержащие:

1) расчет ожидаемой эффективности системы;

2) оценку научно-технического уровня системы.

Приложения включают в состав ТЗ на проектирование ИС по согласованию между разработчиком и заказчиком системы.

Постановка задачи к лабораторной работе 1

1. Изучить предлагаемый теоретический материал.

2. Построить опорные точки зрения на основании метода VORD для формирования и анализа требований. Результатом должны явиться две диаграммы: диаграмма идентификации точек зрения и диаграмма иерархии точек зрения.

3. Составить информационную модель будущей системы, включающую в себя описание основных объектов системы и взаимодействия между ними. На основании информационной модели, диаграммы идентификации точек зрения, диаграммы иерархии точек зрения сформировать требования пользователя и системные требования.

4. На основании описания системы, информационной модели, пользовательских и системных требований разработать техническое задание (ТЗ) на проектирование информационной системы (Приложение А). ТЗ должно содержать основные разделы, описанные в ГОСТ 34.602-89.

5. Оформить отчет по лабораторной работе. Представить отчет по лабораторной работе для защиты.

Варианты индивидуальных заданий

В соответствии с указанной предметной областью и классом разрабатываемой информационной системы разработать техническое задание на проектирование ИС.

Таблица 1.2 – Индивидуальные задания

№	Предметная область	Класс ИС
1	Склад	MRP
2	Производственное предприятие	ERP
3	Торговое предприятие	CRM
4	Торговое предприятие	SCM
5	Торговое предприятие	B2C
6	Портал	B2B
7	Строительное предприятие	ИС учета
8	Высшее учебное заведение	ИС управления
9	Инфраструктура предприятия	СППР
10	Аппаратная инфраструктура предприятия	Экспертная система

Содержание отчета

По выполненной работе составляется отчет. Отчет выполняется в электронном виде. По выполненному отчету проводится защита лабораторной работы.

Отчет по лабораторной работе должен состоять из следующих структурных элементов:

- титульный лист;
- вводная часть;
- основная часть (описание работы): техническое задание на проектирование информационной системы;
- заключение (выводы).

Вводная часть отчета должна включать пункты:
- условие задачи;
- порядок выполнения.
- программно-аппаратные средства, используемые при выполнении работы.

Зашита отчета по лабораторной работе заключается в предъявлении преподавателю полученных результатов в виде файла и демонстрации полученных навыков при ответах на вопросы преподавателя.

Контрольные вопросы

1. Что такое жизненный цикл программного продукта?
2. Дайте определение модели жизненного цикла ПО.
3. Приведите этапы разработки программного средства.
4. Какие этапы включает в себя модель ЖЦ ПС согласно ГОСТ 19.102-77?
5. Что включает в себя этап предпроектного исследования?
6. Перечислите функциональные требования к программному продукту.
7. Перечислите эксплуатационные требования к программному продукту.
8. Перечислите правила разработки технического задания.
9. Назовите основные разделы технического задания.
10. В каких отношениях находятся заказчик и разработчик при выработке требований к программному средству?

Лабораторная работа 2. Применение принципов системной инженерии в проектировании информационных систем. Средства структурного анализа информационных систем.

Цель работы: Изучить и применить на практике методологии структурного анализа информационных систем.

Основы теории

Основные понятия методологии функционального моделирования IDEF0

IDEF0 (Integrated Definition Function Modeling) - методология функционального моделирования. В основе IDEF0 методологии лежит понятие блока, который отображает некоторую бизнес-функцию. Четыре стороны блока имеют разную роль: левая сторона имеет значение "входа", правая - "выхода", верхняя - "управления", нижняя - "механизма" (рисунок 2.1).

Взаимодействие между функциями в IDEF0 представляется в виде дуги, которая отображает поток данных или материалов, поступающий с выхода одной функции на вход другой. В зависимости от того, с какой стороной блока связан поток, его называют соответственно "входным", "выходным", "управляющим".

Рисунок 2.1 - Функциональный блок

Принципы моделирования в IDEF0

В IDEF0 реализованы три базовых принципа моделирования процессов:

принцип функциональной декомпозиции;

принцип ограничения сложности;

принцип контекста.

Принцип функциональной декомпозиции представляет собой способ моделирования типовой ситуации, когда любое действие, операция, функция могут быть разбиты (декомпозированы) на более простые действия, операции, функции. Другими словами, сложная бизнес-функция может быть представлена в виде совокупности элементарных функций. Представляя функции графически, в виде блоков, можно как бы заглянуть внутрь блока и детально рассмотреть ее структуру и состав (рисунок 2.2).

Принцип ограничения сложности. При работе с IDEF0 диаграммами существенным является условие их разборчивости и удобочитаемости. Суть принципа ограничения сложности состоит в том, что количество блоков на диаграмме должно быть не менее двух и не более шести. Практика показывает, что соблюдение этого принципа приводит к тому, что функциональные процессы, представленные в виде IDEF0 модели, хорошо структурированы, понятны и легко поддаются анализу.

Принцип контекстной диаграммы. Моделирование делового процесса начинается с построения контекстной диаграммы. На этой диаграмме отображается только один блок - главная бизнес-функция моделируемой системы. Если речь идет о моделировании целого предприятия или даже крупного подразделения, главная бизнес-функция не может быть сформулирована как, например, "продавать продукцию". Главная бизнес-функция системы - это "миссия" системы, ее значение в окружающем мире. Нельзя правильно сформулировать главную функцию предприятия, не имея представления о его стратегии.

При определении главной бизнес-функции необходимо всегда иметь ввиду цель моделирования и точку зрения на модель. Одно и то же предприятие может быть описано по-разному, в зависимости от того, с какой точки зрения его рассматривают: директор предприятия и налоговой инспектор видят организацию совершенно по-разному.

Контекстная диаграмма играет еще одну роль в функциональной модели. Она "фиксирует" границы моделируемой бизнес-системы, определяя то, как моделируемая система взаимодействует со своим окружением. Это достигается за счет описания дуг, соединенных с блоком, представляющим главную бизнес-функцию.

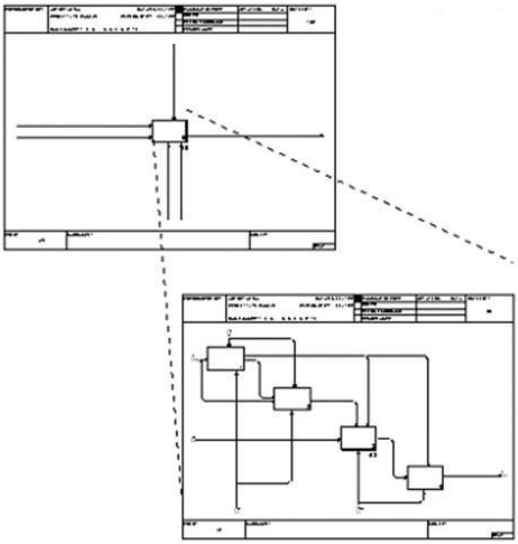

Рисунок 2.2 - Декомпозиция функционального блока

Пример

На рисунках 2.3, 2.4 представлен пример построения функциональной диаграммы, описывающей изготовление изделия. Рисунок 2.3 является примером построения контекстной диаграммы, рисунок 2.4 представляет собой первый уровень декомпозиции.

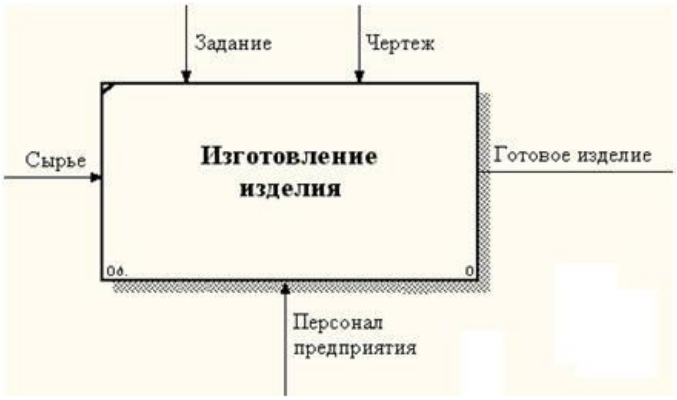

Рисунок 2.3 - Контекстная диаграмма

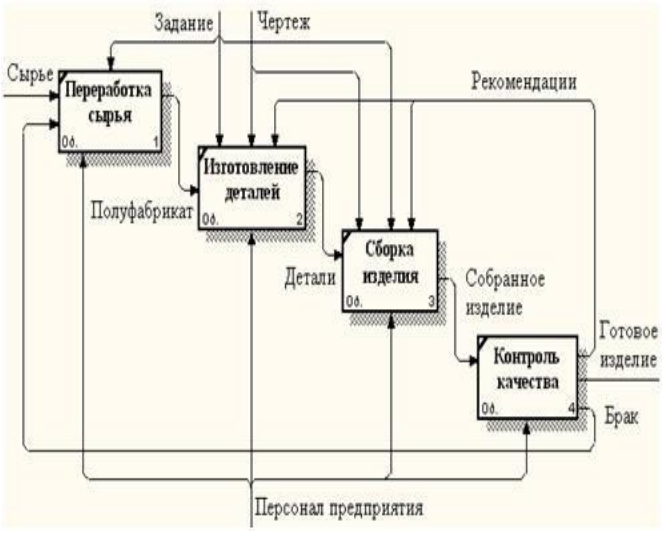

Рисунок 2.4 - Диаграмма первого уровня декомпозиции

Применение IDEF0

Существует два ключевых подхода к построению функциональной модели: построение “как есть” и построение “как будет”.

Построение модели “как есть”. Обследование предприятия является обязательной частью любого проекта создания или развития корпоративной информационной системы.

Построение функциональной модели “как есть” позволяет четко зафиксировать, какие деловые процессы осуществляются на предприятии, какие информационные объекты используются при выполнении деловых процессов и отдельных операций. Функциональная модель “как есть” является отправной точкой для анализа потребностей предприятия, выявления проблем и "узких" мест и разработки проекта совершенствования деловых процессов.

Построение модели “как будет”. Создание и внедрение корпоративной информационной системы приводит к изменению условий выполнения отдельных операций, структуры деловых процессов и предприятия в целом. Это приводит к необходимости изменения системы бизнес-правил, используемых на предприятии, модификации должностных инструкций сотрудников. Функциональная модель “как будет” позволяет уже на стадии проектирования будущей информационной системы определить эти изменения. Применение функциональной модели “как будет” позволяет не только сократить сроки внедрения информационной системы, но также снизить риски, связанные с невосприимчивостью персонала к информационным технологиям.

Метод описания процессов IDEF3

Для описания логики взаимодействия информационных потоков наиболее подходит IDEF3, называемая также workflow diagramming - методологией моделирования, использующая графическое описание информационных потоков, взаимоотношений между процессами обработки информации и объектов, являющихся частью этих процессов. Диаграммы Workflow могут быть использованы в моделировании бизнес-процессов для анализа завершенности процедур обработки информации. С их помощью можно описывать сценарии действий сотрудников организации, например последовательность обработки заказа или события, которые необходимо обработать за конечное время. Каждый сценарий сопровождается описанием процесса и может быть использован для документирования каждой функции.

IDEF3 - это метод, имеющий основной целью дать возможность аналитикам описать ситуацию, когда процессы выполняются в определенной последовательности, а также описать объекты, участвующие совместно в одном процессе,

Техника описания набора данных IDEF3 является частью структурного анализа. В отличие от некоторых методик описаний процессов IDEF3 не ограничивает аналитика чрезмерно жесткими рамками синтаксиса, что может привести к созданию неполных или противоречивых моделей.

IDEF3 может быть также использован как метод создания процессов. IDEF3 дополняет IDEF0 и содержит все необходимое для построения моделей, которые в дальнейшем могут быть использованы для имитационного анализа.

Каждая работа в IDEF3 описывает какой-либо сценарий бизнес-процесса и может являться составляющей другой работы. Поскольку сценарий описывает цель и рамки модели, важно, чтобы работы именовались отглагольным существительным, обозначающим процесс действия, или фразой, содержащей такое существительное.

Точка зрения на модель должна быть задокументирована. Обычно это точка зрения человека, ответственного за работу в целом. Также необходимо задокументировать цель модели - те вопросы, на которые призвана ответить модель.

Диаграммы. Диаграмма является основной единицей описания в IDEF3.

Единицы работы - Unit of Work (UOW). UOW, также называемые работами (activity), являются центральными компонентами модели. В IDEF3 работы изображаются прямоугольниками с прямыми углами и имеют имя, выраженное отглагольным

существительным, обозначающим процесс действия, одиночным или в составе фразы, и номер (идентификатор); другое имя существительное в составе той же фразы обычно отображает основной выход (результат) работы, например, "Изготовление изделия".

Связи. Связи показывают взаимоотношения работ. Все связи в IDEF3 однонаправлены и могут быть направлены куда угодно, но обычно диаграммы IDEF3 стараются построить так, чтобы связи были направлены слева направо. В IDEF3 различают три типа стрелок, изображающих связи, стиль которых устанавливается через меню Edit/Arrow Style:

Старшая (Precedence) - сплошная линия, связывающая единицы работ (UOW), Рисуется слева направо или сверху вниз. Показывает, что работа-источник должна закончиться прежде, чем работа-цель начнется.

Отношения (Relational Link) - пунктирная линия для изображения связей между единицами работ (UOW) а также между единицами работ и объектами ссылок.

Потоки объектов (Object Flow) - стрелка с двумя наконечниками, применяется для описания того факта, что объект используется в двух или более единицах работы, например, когда объект порождается в одной работе и используется в другой.

Старшая связь и поток объектов. Старшая связь показывает, что работа-источник заканчивается ранее, чем начинается работа-цель. Часто результатом работы-источника становится объект, необходимый для запуска работы-цели. В этом случае стрелку, обозначающую объект, изображают с двойным наконечником. Имя стрелки должно ясно идентифицировать отображаемый объект. Поток объектов имеет ту же семантику, что и старшая стрелка.

Перекрестки (Junction). Окончание одной работы может служить сигналом к началу нескольких работ, или же одна работа для своего запуска может ожидать окончания нескольких работ. Перекрестки используются для отображения логики взаимодействия стрелок при слиянии и разветвлении или для отображения множества событий, которые могут или должны быть завершены перед началом следующей работы. Различают перекрестки для слияния (Fan-in Junction) и разветвления (Fan-out Junction) стрелок. Перекресток не может использоваться одновременно для слияния и для разветвления. Для внесения перекрестка служит кнопка в палитре инструментов - добавить в диаграмму перекресток Junction. В диалоге Junction Type Editor необходимо указать тип перекрестка. Смысл каждого типа приведен в таблице 2.1.

Таблица 2.1 - Типы перекрестков

Обозначение	Наименование	Смысл в случае слияния стрелок	Смысл в случае разветвления стрелок
	Asynchronous AND	Все предшествующие процессы должны быть завершены	Все следующие процессы должны быть запущены
	Synchronous AND	Все предшествующие процессы завершены одновременно	Все следующие процессы запускаются одновременно
	Asynchronous OR	Один или несколько предшествующих процессов должны быть завершены	Один или несколько следующих процессов должны быть запущены
	Synchronous OR	Один или несколько предшествующих процессов завершены одновременно	Один или несколько следующих процессов запускаются одновременно

X	XOR (Exclusive OR)	Только один предшествующий процесс завершен	Только один следующий процесс запускается

В отличие от IDEF0 в IDEF3 стрелки могут сливаться и разветвляться только через перекрестки.

Декомпозиция работ. В IDEF3 декомпозиция используется для детализации работ. Методология IDEF3 позволяет декомпозировать работу многократно, т.е. работа может иметь множество дочерних работ. Это позволяет в одной модели описать альтернативные потоки. Возможность множественной декомпозиции предъявляет дополнительные требования к нумерации работ. Так, номер работы состоит из номера родительской работы, версии декомпозиции и собственного номера работы на текущей диаграмме (рисунок 2.5).

Рисунок 2.5 - Номер единицы работы (UOW)

Постановка задачи к лабораторной работе 2

1. Изучить предлагаемый теоретический материал.

2. Построить структурную модель системы, описанной в лабораторной работе 1. Модель должна отвечать всем предъявленным к системе требованиям

3. Построить отчёт, включающий все полученные уровни модели.

4. Оформить отчет по лабораторной работе. Представить отчет по лабораторной работе для защиты.

Варианты индивидуальных заданий

В соответствии с указанной предметной областью и классом разрабатываемой информационной системы разработать структурную модель системы.

Таблица 2.1 – Индивидуальные задания

№	Предметная область	Класс ИС
1	Склад	MRP
2	Производственное предприятие	ERP
3	Торговое предприятие	CRM
4	Торговое предприятие	SCM
5	Торговое предприятие	B2C
6	Портал	B2B
7	Строительное предприятие	ИС учета
8	Высшее учебное заведение	ИС управления
9	Инфраструктура предприятия	СППР
10	Аппаратная инфраструктура предприятия	Экспертная система

Содержание отчета

По выполненной работе составляется отчет. Отчет выполняется в электронном виде. По выполненному отчету проводится защита лабораторной работы.

Отчет по лабораторной работе должен состоять из следующих структурных элементов:

- титульный лист;
- вводная часть;

- основная часть (описание работы): функциональная модель информационной системы и ее описание;

- заключение (выводы).

Вводная часть отчета должна включать пункты:

- условие задачи;

- порядок выполнения.

- программно-аппаратные средства, используемые при выполнении работы.

Зашита отчета по лабораторной работе заключается в предъявлении преподавателю полученных результатов в виде файла и демонстрации полученных навыков при ответах на вопросы преподавателя.

Контрольные вопросы

1. Перечислите основные объекты IDEF0, их описание и назначение.
2. Назовите базовые принципы моделирования в IDEF0.
3. В каких случаях целесообразно применять построение модели “как есть”, а в каких “как будет”?
4. Перечислите основные объекты IDEF3, их описание и назначение.
5. В чём смысл использования перекрёстков в IDEF3?
6. В чём отличия IDEF0 и IDEF3? Когда целесообразней использовать IDEF0, а когда IDEF3?

Лабораторная работа 3. Системный инжиниринг проекта информационной системы

Цель работы: Изучить и применить на практике методологии системного инжиниринга проекта информационной системы.

Основы теории

Проблемы управления программными проектами впервые проявились в 60-х - начале 70-х годов, когда провалились многие большие проекты по разработке программных продуктов. Были зафиксированы задержки в создании ПО, оно было ненадежным, затраты на разработку в несколько раз превосходили первоначальные оценки, созданные программные системы часто имели низкие показатели производительности. Причины провалов коренились в тех подходах, которые использовались в управлении проектами. Применяемая методика была основана на опыте управления техническими проектами и оказалась неэффективной при разработке программного обеспечения.

Важно понимать разницу между профессиональной разработкой ПО и любительским программированием. Необходимость управления программными проектами вытекает из того факта, что процесс создания профессионального ПО всегда является субъектом бюджетной политики организации, где оно разрабатывается, и имеет временные ограничения. Работа руководителя программного проекта по большому счету заключается в том, чтобы гарантировать выполнение этих бюджетных и временных ограничений с учетом бизнес-целей организации относительно разрабатываемого ПО.

Руководители проектов призваны спланировать все этапы разработки программного продукта. Они также должны контролировать ход выполнения работ и соблюдения всех требуемых стандартов. Постоянный контроль за ходом выполнения работ необходим для того, чтобы процесс разработки не выходил за временные и бюджетные ограничения. Хорошее управление не гарантирует успешного завершения проекта, но плохое управление обязательно приведет к его провалу. Это может выразиться в задержке сроков сдачи готового ПО, в превышении сметной стоимости проекта и в несоответствии готового ПО спецификации требований.

Процесс разработки ПО существенно отличается от процессов реализации технических проектов, что порождает определенные сложности в управлении программными проектами:

1. Программный продукт нематериален. Программное обеспечение нематериально, его нельзя увидеть или потрогать. Руководитель программного проекта не видит процесс "роста" разрабатываемого ПО. Он может полагаться только на документацию, которая фиксирует процесс разработки программного продукта.

2. Не существует стандартных процессов разработки ПО. На сегодняшний день не существует четкой зависимости между процессом создания ПО и типом создаваемого программного продукта. Другие технические дисциплины имеют длительную историю, процессы разработки технических изделий многократно опробованы и проверены. Процессы создания большинства технических систем хорошо изучены. Изучением же процессов создания ПО специалисты занимаются только последнее время. Поэтому пока нельзя точно предсказать, на каком этапе процесса разработки ПО могут возникнуть проблемы, угрожающие всему программному проекту.

3. Большие программные проекты - это часто "одноразовые" проекты. Большие программные проекты, как правило, значительно отличаются от проектов, реализованных ранее. Поэтому, чтобы уменьшить неопределенность в планировании проекта, руководители проектов должны обладать очень большим практическим опытом. Но постоянные технологические изменения в компьютерной технике и коммуникационном оборудовании обесценивают предыдущий опыт. Знания и навыки, накопленные опытом, могут не востребоваться в новом проекте.

Перечисленные отличия могут привести к тому, что реализация проекта выйдет из временного графика или превысит бюджетные ассигнования. Программные системы зачастую оказываются новинками как в "идеологическом", так и в техническом плане. Поэтому, предвидя возможные проблемы в реализации программного проекта, следует всегда помнить, что многим из них свойственно выходить за рамки временных и бюджетных ограничений.

Процесс управления разработкой программного обеспечения

Невозможно описать и стандартизировать все работы, выполняемые в проекте по созданию ПО. Эти работы весьма существенно зависят от организации, где выполняется разработка ПО, и от типа создаваемого программного продукта. Но всегда можно выделить следующие:

- Написание предложений по созданию ПО.
- Планирование и составление графика работ по созданию ПО.
- Оценивание стоимости проекта.
- Подбор персонала.
- Контроль за ходом выполнения работ.
- Написание отчетов и представлений.

Первая стадия программного проекта может состоять из написания предложений по реализации этого проекта. Предложения должны содержать описание целей проектов и способов их достижения. Они также обычно включают в себя оценки финансовых и временных затрат на выполнение проекта. При необходимости здесь могут приводиться обоснования для передачи проекта на выполнение сторонней организации или команде разработчиков.

Написание предложений — очень ответственная работа, так как для многих организаций вопрос о том, будет ли проект выполняться самой организацией или разрабатываться по контракту сторонней компанией, является критическим. Не существует каких-либо рекомендаций по написанию предложений, многое здесь зависит от опыт.

На этапе планирования проекта определяются процессы, этапы и полученные на каждом из них результаты, которые должны привести к выполнению проекта. Реализация этого плана приведет к достижению целей проекта. Определение стоимости проекта напрямую связано с его планированием, поскольку здесь оцениваются ресурсы, требующиеся для выполнения плана.

Контроль за ходом выполнения работ (мониторинг проекта) — это непрерывный процесс, продолжающийся в течение всего срока реализации проекта. Руководитель должен постоянно отслеживать ход реализации проекта и сравнивать фактические и плановые показатели выполнения работ с их стоимостью. Хотя многие организации имеют механизмы формального мониторинга работ, опытный руководитель может составить ясную картину о стадии развитии проекта просто путем неформального общения с разработчиками.

Неформальный мониторинг часто помогает обнаружить потенциальные проблемы, которые в явном виде могут обнаружиться позднее. Например, ежедневное обсуждение хода выполнения работ может выявить отдельные недоработки в создаваемом программном продукте. Вместо ожидания отчетов, в которых будет отражен факт "пробуксовки" графика работ, можно обсудить со специалистами намечающиеся программистские проблемы и не допустить срыва графика работ.

В течение реализации проекта обычно происходит несколько формальных контрольных проверок хода выполнения работ по созданию ПО. Такие проверки должны дать общую картину хода реализации проекта в целом и показать, насколько уже разработанная часть ПО соответствует целям проекта.

Время выполнения больших программных проектов может занимать несколько лет. В течение этого времени цели и намерения организации, заказавшей программный проект, могут существенно измениться. Может оказаться, что разрабатываемый программный продукт стал уже ненужным либо исходные требования к создаваемому ПО просто устарели и их необходимо кардинально менять. В такой ситуации руководство организации-разработчика может принять решение о прекращении разработки ПО или об изменении проекта в целом с тем, чтобы учесть изменившиеся цели и намерения организации-заказчика.

Руководители проектов обычно обязаны сами подбирать исполнителей для своих проектов. В идеальном случае профессиональный уровень исполнителей должен соответствовать той работе, которую они будут выполнять в ходе реализации проекта. Однако во многих случаях руководители должны полагаться на команду разработчиков, которая далека от идеальной. Такая ситуация может быть вызвана следующими причинами:

1. Бюджет проекта не позволяет привлечь высококвалифицированный персонал. В таком случае за меньшую плату привлекаются менее квалифицированные специалисты.

2. Бывают ситуации, когда невозможно найти специалистов необходимой квалификации как в самой организации-разработчике, так и вне ее. Например, в организации "лучшие люди" могут быть уже заняты в других проектах.

3. Организация хочет повысить профессиональный уровень своих работников. В этом случае она может привлечь к участию в проекте неопытных или недостаточно квалифицированных работников, чтобы они приобрели необходимый опыт и поучились у более опытных специалистов.

Таким образом, почти всегда подбор специалистов для выполнения проекта имеет определенные ограничения и не является свободным. Вместе с тем необходимо, чтобы хотя бы несколько членов группы разработчиков имели квалификацию и опыт, достаточные для работы над данным проектом. В противном случае невозможно избежать ошибок в разработке ПО.

Руководитель проекта обычно обязан посылать отчеты о ходе его выполнения как заказчику, так и подрядным организациям. Это должны быть краткие документы, основанные на информации, извлекаемой из подробных' отчетов о проекте. В этих отчетах должна быть та информация, которая позволяет четко оценить степень готовности создаваемого программного продукта.

В рамках курса «Системная инженерия» выделены следующие роли в группе по разработке ПО:

- Руководитель – общее руководство проектом, написание документации, общение с заказчиком ПО;

- Системный аналитик – разработка требований (составление технического задания, проекта программного обеспечения);

- Тестер – составление плана тестирования и аттестации готового ПО (продукта), составление сценария тестирования, базовый пример, проведение мероприятий по плану тестирования;

- Разработчик – моделирование компонент программного обеспечения, кодирование.

Постановка задачи к лабораторной работе 3

1. Изучить предлагаемый теоретический материал.

2. Построить модель этапов процесса разработки информационных систем и технологий.

3. Построить отчёт, включающий все полученные уровни модели.

4. Оформить отчет по лабораторной работе. Представить отчет по лабораторной работе для защиты.

Варианты индивидуальных заданий

В соответствии с указанной предметной областью и классом разрабатываемой информационной системы разработать модель этапов процесса разработки информационных систем и технологий.

Таблица 3.1 – Индивидуальные задания

№	Предметная область	Класс ИС
1	Склад	MRP
2	Производственное предприятие	ERP
3	Торговое предприятие	CRM
4	Торговое предприятие	SCM
5	Торговое предприятие	B2C
6	Портал	B2B
7	Строительное предприятие	ИС учета
8	Высшее учебное заведение	ИС управления
9	Инфраструктура предприятия	СППР
10	Аппаратная инфраструктура предприятия	Экспертная система

Содержание отчета

По выполненной работе составляется отчет. Отчет выполняется в электронном виде. По выполненному отчету проводится защита лабораторной работы.

Отчет по лабораторной работе должен состоять из следующих структурных элементов:

- титульный лист;

- вводная часть;

- основная часть (описание работы): функциональная модель информационной системы и ее описание;

- заключение (выводы).

Вводная часть отчета должна включать пункты:

- условие задачи;

- порядок выполнения.

- программно-аппаратные средства, используемые при выполнении работы.

Зашита отчета по лабораторной работе заключается в предъявлении преподавателю полученных результатов в виде файла и демонстрации полученных навыков при ответах на вопросы преподавателя.

Контрольные вопросы

1. Перечислите основные этапов процесса разработки информационных систем и технологий.
2. Назовите базовые принципы процесса разработки информационных систем.
3. Перечислите модели жизненного цикла разработки ПО.
4. Опишите инструменты инжиниринга проекта информационной системы
5. Охарактеризуйте методы реинжиниринга проекта информационной системы

Лабораторная работа 4. Инжиниринг и реинжиниринг архитектуры информационных систем. Спецификация программного обеспечения при объектном подходе.

Цель работы: Изучить и применить на практике методологии инжиниринга и реинжиниринга архитектуры информационных систем, методы разработки спецификации программного обеспечения при объектном подходе.

Основы теории

Планирование проекта разработки программного обеспечения

Эффективное управление программным проектом напрямую зависит от правильного планирования работ, необходимых для его выполнения. План помогает руководителю предвидеть проблемы, которые могут возникнуть на каких-либо этапах создания ПО, и разработать превентивные меры для их предупреждения или решения. План, разработанный на начальном этапе проекта, рассматривается всеми его участниками как руководящий документ, выполнение которого должно привести к успешному завершению проекта. Этот первоначальный план должен максимально подробно описывать все этапы реализации проекта.

Процесс планирования начинается, исходя из описания системы, с определения проектных ограничений (временные ограничения, возможности наличного персонала, бюджетные ограничения и т.д.). Эти ограничения должны определяться параллельно с оцениванием проектных параметров, таких как структура и размер проекта, а также распределением функций среди исполнителей. Затем определяются этапы разработки и то, какие результаты документация, прототипы, подсистемы или версии программного продукта) должны быть получены по окончании этих этапов. Далее начинается циклическая часть планирования. Сначала разрабатывается график работ по выполнению проекта или дается разрешение на продолжение использования ранее созданного графика. После этого проводится контроль выполнения работ и отмечаются расхождения между реальным и плановым ходом работ.

Далее, по мере поступления новой информации о ходе выполнения проекта, возможен пересмотр первоначальных оценок параметров проекта. Это, в свою очередь, может привести к изменению графика работ. Если в результате этих изменений нарушаются сроки завершения проекта, должны быть пересмотрены (и согласованы с заказчиком ПО) проектные ограничения.

Конечно, большинство руководителей проектов не думают, что реализация их проектов пройдет гладко, без всяких проблем. Желательно описать возможные проблемы еще до того, как они проявят себя в ходе выполнения проекта. Поэтому лучше составлять "пессимистические" графики работ, чем "оптимистические". Но, конечно, невозможно построить план, учитывающий все, в том числе случайные, проблемы и задержки выполнения проекта, поэтому и возникает необходимость периодического пересмотра проектных ограничений и этапов создания программного продукта.

План проекта должен четко показать ресурсы, необходимые для реализации проекта, разделение работ на этапы и временной график выполнения этих этапов. В некоторых организациях план проекта составляется как единый документ, содержащий все виды планов, описанных выше. В других случаях план проекта описывает только технологический процесс создания ПО. В таком плане обязательно присутствуют ссылки на планы других видов, но они разрабатываются отдельно от плана проекта.

Детализация планов проектов очень разнится в зависимости от типа разрабатываемого программного продукта и организации-разработчика. Но в любом случае большинство планов содержат следующие разделы.

1. Введение. Краткое описание целей проекта и проектных ограничений (бюджетных, временных и т.д.), которые важны для управления проектом.

2. Организация выполнения проекта. Описание способа подбора команды разработчиков и распределение обязанностей между членами команды.

3. Анализ рисков. Описание возможных проектных рисков, вероятности их проявления и стратегий, направленных на их уменьшение.

4. Аппаратные и программные ресурсы, необходимые для реализации проекта. Перечень аппаратных средств и программного обеспечения, необходимого для разработки программного продукта. Если аппаратные средства требуется закупать, приводится их стоимость совместно с графиком закупки и поставки.

5. Разбиение работ на этапы. Процесс реализации проекта разбивается на отдельные процессы, определяются этапы выполнения проекта, приводится описание результатов ("выходов") каждого этапа и контрольные отметки.

6. График работ. В этом графике отображаются зависимости между отдельными процессами (этапами) разработки ПО, оценки времени их выполнения и распределение членов команды разработчиков по отдельным этапам.

7. Механизмы мониторинга и контроля за ходом выполнения проекта. Описываются предоставляемые руководителем отчеты о ходе выполнения работ, сроки их предоставления, а также механизмы мониторинга всего проекта.

План должен регулярно пересматриваться в процессе реализации проекта. Одни части плана, например график работ, изменяются часто, другие более стабильны. Для внесения изменений в план требуется специальная организация документопотока, позволяющая отслеживать эти изменения.

Постановка задачи к лабораторной работе 4

1. Изучить предлагаемый теоретический материал.

2. Провести объектно-ориентированный анализ проекта информационной системы, описанной в лабораторной работе 1.

3. Выполните реализацию вариантов использования в терминах взаимодействующих объектов и представляющую собой набор диаграмм:

3.1 диаграмм классов, реализующих вариант использования;

3.2 диаграмм взаимодействия (диаграмм последовательности и кооперативных диаграмм), отражающих взаимодействие объектов в процессе реализации варианта использования.

4. Разделить классы по пакетам, используя один из механизм разбиения.

5. Построить отчёт, включающий все полученные уровни модели. Представить отчет для защиты.

Варианты индивидуальных заданий

В соответствии с указанной предметной областью и классом разрабатываемой информационной системы провести объектно-ориентированный анализ проекта информационной системы.

Таблица 4.1 – Индивидуальные задания

№	Предметная область	Класс ИС
1	Склад	MRP
2	Производственное предприятие	ERP
3	Торговое предприятие	CRM
4	Торговое предприятие	SCM
5	Торговое предприятие	B2C
6	Портал	B2B
7	Строительное предприятие	ИС учета
8	Высшее учебное заведение	ИС управления
9	Инфраструктура предприятия	СППР
10	Аппаратная инфраструктура предприятия	Экспертная система

Содержание отчета

По выполненной работе составляется отчет. Отчет выполняется в электронном виде. По выполненному отчету проводится защита лабораторной работы.

Отчет по лабораторной работе должен состоять из следующих структурных элементов:

- титульный лист;
- вводная часть;
- основная часть (описание работы): функциональная модель информационной системы и ее описание;
- заключение (выводы).

Вводная часть отчета должна включать пункты:

- условие задачи;
- порядок выполнения.
- программно-аппаратные средства, используемые при выполнении работы.

Зашита отчета по лабораторной работе заключается в предъявлении преподавателю полученных результатов в виде файла и демонстрации полученных навыков при ответах на вопросы преподавателя.

Контрольные вопросы

1. Перечислите основные этапы инжиниринга и реинжиниринга архитектуры информационных систем.
2. Назовите базовые принципы моделирования архитектуры информационных систем.
3. В каких случаях целесообразно применять функциональный и объектно-ориентированный анализ проекта информационной системы?
4. Назовите инструменты функционального и объектно-ориентированного анализа проекта информационной системы?

6. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

6.1. Перечень основной и дополнительной литературы, необходимой для освоения дисциплины

6.1.1. Перечень основной литературы

1. Системная инженерия: учебное пособие/ Министерство образования и науки Российской Федерации, Федеральное государственное автономное образовательное учреждение высшего профессионального образования «Северо-Кавказский федеральный

университет»; авт.-сост. В.И.Дроздова. - Ставрополь: СКФУ, 2014. - 115 с. - То же [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page= book&id =458082.

2. Маглинец Ю.А. Анализ требований к автоматизированным информационным системам [Электронный ресурс]/ Маглинец Ю.А.— Электрон. текстовые данные.— М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016.— 191 с.— Режим доступа: http://www.iprbookshop.ru/52184.— ЭБС «IPRbooks», по паролю

6.1.2. Перечень дополнительной литературы

1. Лиманова Н.И. Архитектура вычислительных систем и компьютерных сетей [Электронный ресурс]: учебное пособие/ Н.И. Лиманова. — Электрон. текстовые данные. — Самара: Поволжский государственный университет телекоммуникаций и информатики, 2017. — 197 с. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru /75368.html.

2. Клименко И.С. Теория систем и системный анализ [Электронный ресурс]: учебное пособие / И.С. Клименко. — Электрон. текстовые данные. — М.: Российский новый университет, 2014. — 264 с. —Режим доступа: http://www.iprbookshop.ru/21322.html.

6.2. Перечень учебно-методического обеспечения самостоятельной работы обучающихся по дисциплине

1. Методические указания по выполнению лабораторных работ по дисциплине «Системная инженерия».

2. Методические рекомендации для студентов по организации самостоятельной работы по дисциплине «Системная инженерия».

6.3. Перечень ресурсов информационно-телекоммуникационной сети Интернет, необходимых для освоения дисциплины

1. Университетская библиотека online. http://www.biblioclub.ru.

2. ЭБС «IPRbooks». http://www.iprbookshop.ru.

3. Электронная библиотека СКФУ.. http://catalog.ncstu.ru.

4. Государственная публичная научно- техническая библиотека России. (ГПНТБ России). www.gpntb.ru.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Пятигорский институт (филиал)

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ СТУДЕНТОВ ПО ОРГАНИЗАЦИИ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ПО ДИСЦИПЛИНЕ СИСТЕМНАЯ ИНЖЕНЕРИЯ

Направление подготовки	**09.04.02 Информационные системы и технологии**
Направленность (профиль)	**Технологии работы с данными и знаниями, анализ информации**
Квалификация выпускника	Магистр

Пятигорск, 2025

СОДЕРЖАНИЕ

Введение....33
1. Цель и задачи изучения дисциплины....35
2. Темы самостоятельной работы....35
3. Технологическая карта самостоятельной работы обучающегося....35
4. Рекомендации для самоподготовки....35
4.1 Подготовка к лекциям. Самостоятельное изучение литературы....36
4.2 Подготовка к лабораторным работам....38
4.3 Материалы для самостоятельного изучения....39
Методы и средства системной инженерии....39
Системный инжиниринг проекта информационной системы....44
5. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ....48
5.1. Перечень основной и дополнительной литературы, необходимой для освоения дисциплины....48
5.2. Перечень учебно-методического обеспечения самостоятельной работы обучающихся по дисциплине....48
5.3. Перечень ресурсов информационно-телекоммуникационной сети Интернет, необходимых для освоения дисциплины....49

ВВЕДЕНИЕ

Методические рекомендации содержат перечень тем с вопросами для самостоятельной проработки, перечень практических и лабораторных работ с вопросами для самостоятельной проработки, материал для подготовки контрольной работы и курсового проекта.

Методические рекомендации посвящены курсу «Системная инженерия». Самое современное определение системной инженерии дано в Guide to the Systems Engineering Body of Knowledge (руководство по корпусу знаний системной инженерии). Короткое определение понятия системная инженерия: это междисциплинарный подход и способы обеспечения воплощения успешной системы (Systems engineering is an interdisciplinary approach and means to enable the realization of successful systems). В этом определении можно подчеркнуть следующие моменты:

1. Успешные системы — это то, чем занимается системная инженерия. Слово “успешные” тут крайне важно и означает, что система должна удовлетворить нужды заказчиков, пользователей и других стейкхолдеров (стейкхолдеры — это те, кто затрагивается системой, или кто затрагивает систему). Успех —э то когда системой все довольны.

2. Слово “системы” используется в очень специальном значении: это «системы» из системного подхода. Если вы сказали «система» про компьютер, то это автоматически влечёт за собой разговор про стейкхолдеров и их интересы, требования и архитектуру, жизненный цикл и т.д.

3. Междисциплинарный подход — системная инженерия претендует на то, что она работает со всеми остальными инженерными специальностями (впрочем, не только инженерными). «Подход» обычно означает какие-то наработки в одной предметной области, которые можно перенести на другие предметные области. Междисциплинарность — это очень сильное заявление, оно означает, что системная инженерия может в одну упряжку впрячь коня и трепетную лань (например, инженеров-механиков, баллистиков, криогенщиков, психологов, медиков, астрономов, программистов и т.д. в проектах пилотируемой космонавтики).

4. Слово “воплощение” (realization, “перевод в реальность”) означает буквально это: создание материальной (из вещества и полей) успешной системы.

По-английски “системная инженерия” — systems engineering, хотя более ранние написания были как system engineering. Правильная интерпретация (и правильный перевод) — именно «системная» (подразумевающая использование системного подхода) инженерия, а не «инженерия систем» (engineering of systems) — когда любой «объект» называется «системой», но не используется системный подход во всей его полноте.

Из системной инженерии квалификатор "системный" без изменения смысла понятия выкинуть нельзя — системная инженерия это инженерия с системным мышлением.

Более длинное определение включает ещё одну фразу: «Она фокусируется на целостном и одновременном/параллельном понимании нужд стейкхолдеров; исследовании возможностей; документировании требований; и синтезировании, проверке, приёмке и постепенном появлении инженерных решений, в то время как в расчёт принимается полная проблема, от исследования концепции системы до вывода системы из эксплуатации» (It focuses on holistically and concurrently understanding stakeholder needs; exploring opportunities; documenting requirements; and synthesizing, verifying, validating, and evolving solutions while considering the complete problem, from system concept exploration through system disposal). Тут нужно подчеркнуть:

1 Целокупность, которая подчёркнута многократно — от “междисциплинарности” в первой половине определения до целостности всех действий по созданию системы во второй половине определения, до целостности/полноты проблемы, до охвата всего

жизненного цикла системы “от рождения до смерти”. Целостность (полнота охвата всех частей целевой системы согласованным их целым), междисциплинарность (полнота охвата всех дисциплин) — это ключевое, что отличает системную инженерию от всех остальных инженерных дисциплин.

2. Параллельность выполнения самых разных практик (а не последовательное выполнение их во времени, как можно было бы подумать, прочитав перечисление практик)

3. Много особенностей (различение нужд пользователей и требований, проверки и приёмки, упор на синтез для противопоставления “аналитическим” дисциплинам и т.д.).

Ответственность за всю систему как целое (whole system) и связанная с этим межпредметность/междисциплинарность (interdisciplinary) подхода к другим инженериям (механической, электрической, программной, предприятия и т.д.) отличают системную инженерию от всех других инженерных дисциплин.

Представим себе ледовую буровую платформу: Сотни тысяч тонн металла, бетона, пластмассы, необходимых расходных материалов, обученная вахта должны собраться вместе далеко в море среди льда и в строго определённый момент эта огромная конструкция должна начать согласованно работать — и не просто работать, а приносить прибыль и обеспечивать безопасность в части загрязнения окружающей среды и здоровья находящейся на платформе вахты. Какая инженерная дисциплина должна учесть результаты работ всех других инженерных дисциплин — собрать в единое целое данные ледовой обстановки, санитарных норм в помещениях для обслуживающего персонала, обеспечение электричеством попавших туда компьютерных серверов, характеристики эти серверов и программное обеспечение? Кто озаботится учётом в конструкции платформы изменений в длине металлоконструкций за счёт разницы суточных температур и одновременно установкой акустических датчиков на трубах, которые прослушивают шорох песка, чтобы по этому шороху можно было определить износ труб?

Системная инженерия как раз и является той дисциплиной, которая ответственна за обеспечение целостности в инженерном проекте: именно системные инженеры проектируют нефтяную платформу как успешное (безопасное, надёжное, прибыльное, ремонтопригодное и т.д. — разным людям нужно от этой платформы разное) целое, потом раздают части работы инженерам по специальностям (инженерам-строителям, машиностроителям, инженерам-электрикам, компьютерщикам/айтишникам и т.д.), а затем собирают результаты их работ так, чтобы получить работоспособную и надёжную систему.

Это главный признак, отличающий системных инженеров от всех других инженеров: они отвечают за проект в целом, сразу во всех его деталях как в части деталей-частей, так и в части использования детальных знаний отдельных дисциплин. Они ответственны за то, чтобы не было пропущено какой-нибудь мелочи, ведущей к провалу.

Самолёт — это много-много кусков металла и пластика, синхронно летящих на скорости 900км/час (0.85 от скорости звука, это типовая скорость Boeing 787 Dreamliner) на высоте 10км. Системный инженер — это тот, кто придумал, как обеспечить их надёжный и экономичный совместный полёт, увязав самые разные требования (грузоподъёмность, расход топлива, дальность полёта, шум при взлёте и посадке, требования к длине разбега и посадки, необходимость лёгкого обслуживания на земле, отсутствие обледенения, безопасность людей на борту, и т.д. и т.п.), при этом требования выдвигались самыми разными людьми, представляющими самые разные профессиональные и общественные группы. Пара-тройка миллионов деталей изготавливается и собирается в одно изделие — и самолёт летит, обеспечивая комфорт пассажирам и прибыль владельцам.

Системная инженерия решает задачу преодоления инженерной сложности (которая определяется главным образом как число различных элементов, которые должны провзаимодействовать друг с другом, чтобы получить целевую систему — и сложные системы не помещаются ни в какую самую умную голову, требуется специальная дисциплина, чтобы собирать результаты деятельности самых разных инженеров в одно работоспособное целое.

1. ЦЕЛЬ И ЗАДАЧИ ИЗУЧЕНИЯ ДИСЦИПЛИНЫ

Целью освоения дисциплины «Системная инженерия» является формирование набора профессиональных компетенций будущего магистра по направлению подготовки 09.04.02 «Информационные системы и технологии» для решения прикладных задач в рамках магистерской программы «Технологии работы с данными и знаниями, анализ информации».

Задачи освоения дисциплины: изучение основных понятий системной инженерии, освоение методов и инструментов системной инженерии.

2. ТЕМЫ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

№ темы	Наименование тем дисциплины, их краткое содержание	Обьем часов
	2 семестр	
	Раздел 1. Системная инженерия процессов разработки информационных систем и технологий	
1	Тема 1. Методы и средства системной инженерии Методы и средства системной инженерии. Технологии системной инженерии. Методологии работы с данными и знаниями.	50
	Итого за 2 семестр	50
	3 семестр	
2	Тема 2. Инструментальное обеспечение системной инженерии Инструментальное обеспечение системной инженерии. Функциональные и технологические требования к информационным системам.	50
3	Тема 5. Инжиниринг и реинжиниринг архитектуры информационных систем.	90,25
	Итого за 3 семестр	140,25
	Итого	190,25

3. ТЕХНОЛОГИЧЕСКАЯ КАРТА САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩЕГОСЯ

Для студентов заочной формы обучения:

Коды реализуемых компетенций, индикатора(ов)	Вид деятельности студентов	Средства и технологии оценки	Объем часов, в том числе		
			СРС	Контактная работа с преподавателем	Всего

2 семестр					
УК-2, УК-3, ОПК-5, ОПК-7, ОПК-8	Подготовка к лекциям	Собеседование	0,18	0,02	0,2
УК-2, УК-3, ОПК-5, ОПК-7, ОПК-8	Самостоятельное изучение литературы по теме 1	Собеседование	60,48	6,72	67,2
УК-2, УК-3, ОПК-5, ОПК-7, ОПК-8	Подготовка к лабораторным работам	Отчет письменный	0,54	0,06	0,6
Итого за 2 семестр			61,2	6,8	68
3 семестр					
УК-2, УК-3, ОПК-5, ОПК-7, ОПК-8	Подготовка к лекциям	Собеседование	0,72	0,08	0,8
УК-2, УК-3, ОПК-5, ОПК-7, ОПК-8	Самостоятельное изучение литературы по темам 2, 5	Собеседование	155,34	17,26	172,6
УК-2, УК-3, ОПК-5, ОПК-7, ОПК-8	Подготовка к лабораторным работам	Отчет письменный	3,24	0,36	3,6
УК-2, УК-3, ОПК-5, ОПК-7, ОПК-8	Подготовка к экзамену	Устный экзамен	9	1	10
Итого за 3 семестр			168,3	18,7	187
Итого			168,3	18,7	187

4. РЕКОМЕНДАЦИИ ДЛЯ САМОПОДГОТОВКИ

4.1 Подготовка к лекциям. Самостоятельное изучение литературы

Тема 1. Методы и средства системной инженерии

Базовый уровень

1. Рабочий проект информационной системы. Дисциплина обязательств.
2. Технический проект информационной системы
3. Управление проектом информационной системы
4. Календарный план разработки информационной системы.
5. Понятие архитектуры информационной системы

Повышенный уровень

1. Применение инструментов разработки информационных систем. Язык программирования, блок-схема, документирование, средства разработки алгоритмическая, эффективность алгоритма
2. Управление требованиями к информационной системе
3. Виды требований к информационной системе: функциональные требования, нефункциональные требования.
4. Свойства требований к информационной системе: ясность и недвусмысленность, полнота и непротиворечивость, необходимый уровень детализации, прослеживаемость, тестируемость и проверяемость, модифицируемость.
5. Формализация требований к информационной системе. Цикл работы с требованиями

Тема 2. Инструментальное обеспечение системной инженерии

Базовый уровень

1. Открытая архитектура информационных систем
2. Системная инженерия: точка зрения и характеристики точек зрения
3. Множественность точек зрения при разработке информационной системы
4. Характеристики информационной системы. Интерактивность, usability

5. Применение инструментов моделирования информационных систем.

Повышенный уровень

1. Понятие конфигурационного управления проектом информационной системы
2. Управление версиями информационной системы. Понятие "ветки" проекта
3. Управление сборками при разработке информационной системы
4. Средства версионного контроля информационной системы
5. Единицы конфигурационного управления. Понятие baseline.

Тема 3. Применение принципов системной инженерии в проектировании информационных систем

Базовый уровень

1. Принципы верификации и тестирования
2. Верификация и валидация программных продуктов
3. Понятие тестирования программных средств
4. Методы верификации объектно-ориентированных программ
5. Качество и надежность программного обеспечения

Повышенный уровень

1. Функциональные и технологические требования к программным комплексам
2. Архитектура программных комплексов для информатизации предприятий,
3. Стандарты проектирования программного обеспечения
4. Стандарты разработки программного обеспечения
5. Методы разработки программных комплексов для решения прикладных задач

Тема 4. Системный инжиниринг проекта информационной системы

Базовый уровень

1. Профили открытых информационных систем
2. Функциональные и технологические стандарты разработки программных комплексов
3. Принципы организации проектирования и программных комплексов
4. Задачи обеспечения качества программных компонентов
5. Методы исследования качества программных компонентов

Повышенный уровень

1. Методы оценки сложности алгоритмов и программ
2. Методы тестирования и документирования программных комплексов
3. Основные составные информационной системы и их характеристика.
4. Многоуровневая архитектура информационных систем
5. Моделирование структуры информационных систем, виды моделей, их назначение

Тема 5. Инжиниринг и реинжиниринг архитектуры информационных систем

Базовый уровень

1 Задачи обеспечения надежности программных компонентов
2. Методы исследования надежности программных компонентов
3. Экономико-правовые основы разработки программных продуктов
4. Объектно-ориентированное моделирование, диаграммы вариантов использования.
5. Многопользовательская информационная система с распределенной базой данных
6. Рабочий проект информационной системы. Дисциплина обязательств.
7. Технический проект информационной системы
8. Управление проектом информационной системы
9. Календарный план разработки информационной системы.
10. Понятие архитектуры информационной системы

Повышенный уровень

1. Метрики качества программного обеспечения

2. Стандартный метод оценки значений показателей качества
3. Управление качеством ПО
4. Оптимизация и реинжиниринг
5. Инновационные проекты.
6. Тестирование информационной системы
7. Стандартизация качества информационных систем
8. Методы обеспечения качества информационных систем
9. Понятие тестирования информационной системы. Тестирование черного ящика. Тестирование белого ящика
10. Инструменты тестирования информационных систем

4.2 Подготовка к лабораторным работам

Лабораторная работа 1. Применение принципов системной инженерии в проектировании информационных систем. Разработка технического задания на проектирование информационной системы

1. Что такое жизненный цикл программного продукта?
2. Дайте определение модели жизненного цикла ПО.
3. Приведите этапы разработки программного средства.
4. Какие этапы включает в себя модель ЖЦ ПС согласно ГОСТ 19.102-77?
5. Что включает в себя этап предпроектного исследования?
6. Перечислите функциональные требования к программному продукту.
7. Перечислите эксплуатационные требования к программному продукту.
8. Перечислите правила разработки технического задания.
9. Назовите основные разделы технического задания.
10. В каких отношениях находятся заказчик и разработчик при выработке требований к программному средству?

Лабораторная работа 2. Применение принципов системной инженерии в проектировании информационных систем. Средства структурного анализа информационных систем.

1. Перечислите основные объекты IDEF0, их описание и назначение.
2. Назовите базовые принципы моделирования в IDEF0.
3. В каких случаях целесообразно применять построение модели “как есть”, а в каких “как будет”?
4. Перечислите основные объекты IDEF3, их описание и назначение.
5. В чём смысл использования перекрёстков в IDEF3?
6. В чём отличия IDEF0 и IDEF3? Когда целесообразней использовать IDEF0, а когда IDEF3?

Лабораторная работа 3. Системный инжиниринг проекта информационной системы

1. Перечислите основные этапов процесса разработки информационных систем и технологий.
2. Назовите базовые принципы процесса разработки информационных систем.
3. Перечислите модели жизненного цикла разработки ПО.
4. Опишите инструменты инжиниринга проекта информационной системы
5. Охарактеризуйте методы реинжиниринга проекта информационной системы

Лабораторная работа 4. Инжиниринг и реинжиниринг архитектуры информационных систем. Спецификация программного обеспечения при объектном подходе.

1. Перечислите основные этапы инжиниринга и реинжиниринга архитектуры информационных систем.

2. Назовите базовые принципы моделирования архитектуры информационных систем.

3. В каких случаях целесообразно применять функциональный и объектно-ориентированный анализ проекта информационной системы?

4. Назовите инструменты функционального и объектно-ориентированного анализа проекта информационной системы?

4.3 Материалы для самостоятельного изучения

Для выполнения самостоятельной работы следует изучить теоретический материал.

МЕТОДЫ И СРЕДСТВА СИСТЕМНОЙ ИНЖЕНЕРИИ

Проблемы управления программными проектами впервые проявились в 60-х - начале 70-х годов, когда провалились многие большие проекты по разработке программных продуктов. Были зафиксированы задержки в создании ПО, оно было ненадежным, затраты на разработку в несколько раз превосходили первоначальные оценки, созданные программные системы часто имели низкие показатели производительности. Причины провалов коренились в тех подходах, которые использовались в управлении проектами. Применяемая методика была основана на опыте управления техническими проектами и оказалась неэффективной при разработке программного обеспечения.

Важно понимать разницу между профессиональной разработкой ПО и любительским программированием. Необходимость управления программными проектами вытекает из того факта, что процесс создания профессионального ПО всегда является субъектом бюджетной политики организации, где оно разрабатывается, и имеет временные ограничения. Работа руководителя программного проекта по большому счету заключается в том, чтобы гарантировать выполнение этих бюджетных и временных ограничений с учетом бизнес-целей организации относительно разрабатываемого ПО.

Руководители проектов призваны спланировать все этапы разработки программного продукта. Они также должны контролировать ход выполнения работ и соблюдения всех требуемых стандартов. Постоянный контроль за ходом выполнения работ необходим для того, чтобы процесс разработки не выходил за временные и бюджетные ограничения. Хорошее управление не гарантирует успешного завершения проекта, но плохое управление обязательно приведет к его провалу. Это может выразиться в задержке сроков сдачи готового ПО, в превышении сметной стоимости проекта и в несоответствии готового ПО спецификации требований.

Процесс разработки ПО существенно отличается от процессов реализации технических проектов, что порождает определенные сложности в управлении программными проектами:

1. Программный продукт нематериален. Программное обеспечение нематериально, его нельзя увидеть или потрогать. Руководитель программного проекта не видит процесс "роста" разрабатываемого ПО. Он может полагаться только на документацию, которая фиксирует процесс разработки программного продукта.

2. Не существует стандартных процессов разработки ПО. На сегодняшний день не существует четкой зависимости между процессом создания ПО и типом создаваемого программного продукта. Другие технические дисциплины имеют длительную историю, процессы разработки технических изделий многократно опробованы и проверены. Процессы создания большинства технических систем хорошо изучены. Изучением же процессов создания ПО специалисты занимаются только последнее время. Поэтому пока нельзя точно предсказать, на каком этапе процесса разработки ПО могут возникнуть проблемы, угрожающие всему программному проекту.

3. Большие программные проекты - это часто "одноразовые" проекты. Большие программные проекты, как правило, значительно отличаются от проектов, реализованных ранее. Поэтому, чтобы уменьшить неопределенность в планировании проекта,

руководители проектов должны обладать очень большим практическим опытом. Но постоянные технологические изменения в компьютерной технике и коммуникационном оборудовании обесценивают предыдущий опыт. Знания и навыки, накопленные опытом, могут не востребоваться в новом проекте.

Перечисленные отличия могут привести к тому, что реализация проекта выйдет из временного графика или превысит бюджетные ассигнования. Программные системы зачастую оказываются новинками как в "идеологическом", так и в техническом плане. Поэтому, предвидя возможные проблемы в реализации программного проекта, следует всегда помнить, что многим из них свойственно выходить за рамки временных и бюджетных ограничений.

Процесс управления разработкой программного обеспечения

Невозможно описать и стандартизировать все работы, выполняемые в проекте по созданию ПО. Эти работы весьма существенно зависят от организации, где выполняется разработка ПО, и от типа создаваемого программного продукта. Но всегда можно выделить следующие:

- Написание предложений по созданию ПО.
- Планирование и составление графика работ по созданию ПО.
- Оценивание стоимости проекта.
- Подбор персонала.
- Контроль за ходом выполнения работ.
- Написание отчетов и представлений.

Первая стадия программного проекта может состоять из написания предложений по реализации этого проекта. Предложения должны содержать описание целей проектов и способов их достижения. Они также обычно включают в себя оценки финансовых и временных затрат на выполнение проекта. При необходимости здесь могут приводиться обоснования для передачи проекта на выполнение сторонней организации или команде разработчиков.

Написание предложений — очень ответственная работа, так как для многих организаций вопрос о том, будет ли проект выполняться самой организацией или разрабатываться по контракту сторонней компанией, является критическим. Не существует каких-либо рекомендаций по написанию предложений, многое здесь зависит от опыт.

На этапе планирования проекта определяются процессы, этапы и полученные на каждом из них результаты, которые должны привести к выполнению проекта. Реализация этого плана приведет к достижению целей проекта. Определение стоимости проекта напрямую связано с его планированием, поскольку здесь оцениваются ресурсы, требующиеся для выполнения плана.

Контроль за ходом выполнения работ (мониторинг проекта) — это непрерывный процесс, продолжающийся в течение всего срока реализации проекта. Руководитель должен постоянно отслеживать ход реализации проекта и сравнивать фактические и плановые показатели выполнения работ с их стоимостью. Хотя многие организации имеют механизмы формального мониторинга работ, опытный руководитель может составить ясную картину о стадии развитии проекта просто путем неформального общения с разработчиками.

Неформальный мониторинг часто помогает обнаружить потенциальные проблемы, которые в явном виде могут обнаружиться позднее. Например, ежедневное обсуждение хода выполнения работ может выявить отдельные недоработки в создаваемом программном продукте. Вместо ожидания отчетов, в которых будет отражен факт "пробуксовки" графика работ, можно обсудить со специалистами намечающиеся программистские проблемы и не допустить срыва графика работ.

В течение реализации проекта обычно происходит несколько формальных контрольных проверок хода выполнения работ по созданию ПО. Такие проверки должны дать общую картину хода реализации проекта в целом и показать, насколько уже разработанная часть ПО соответствует целям проекта.

Время выполнения больших программных проектов может занимать несколько лет. В течение этого времени цели и намерения организации, заказавшей программный проект, могут существенно измениться. Может оказаться, что разрабатываемый программный продукт стал уже ненужным либо исходные требования к создаваемому ПО просто устарели и их необходимо кардинально менять. В такой ситуации руководство организации-разработчика может принять решение о прекращении разработки ПО или об изменении проекта в целом с тем, чтобы учесть изменившиеся цели и намерения организации-заказчика.

Руководители проектов обычно обязаны сами подбирать исполнителей для своих проектов. В идеальном случае профессиональный уровень исполнителей должен соответствовать той работе, которую они будут выполнять в ходе реализации проекта. Однако во многих случаях руководители должны полагаться на команду разработчиков, которая далека от идеальной. Такая ситуация может быть вызвана следующими причинами:

1. Бюджет проекта не позволяет привлечь высококвалифицированный персонал. В таком случае за меньшую плату привлекаются менее квалифицированные специалисты.

2. Бывают ситуации, когда невозможно найти специалистов необходимой квалификации как в самой организации-разработчике, так и вне ее. Например, в организации "лучшие люди" могут быть уже заняты в других проектах.

3. Организация хочет повысить профессиональный уровень своих работников. В этом случае она может привлечь к участию в проекте неопытных или недостаточно квалифицированных работников, чтобы они приобрели необходимый опыт и поучились у более опытных специалистов.

Таким образом, почти всегда подбор специалистов для выполнения проекта имеет определенные ограничения и не является свободным. Вместе с тем необходимо, чтобы хотя бы несколько членов группы разработчиков имели квалификацию и опыт, достаточные для работы над данным проектом. В противном случае невозможно избежать ошибок в разработке ПО.

Руководитель проекта обычно обязан посылать отчеты о ходе его выполнения как заказчику, так и подрядным организациям. Это должны быть краткие документы, основанные на информации, извлекаемой из подробных' отчетов о проекте. В этих отчетах должна быть та информация, которая позволяет четко оценить степень готовности создаваемого программного продукта.

В рамках курса «Системная инженерия» выделены следующие роли в группе по разработке ПО:

- Руководитель – общее руководство проектом, написание документации, общение с заказчиком ПО;
- Системный аналитик – разработка требований (составление технического задания, проекта программного обеспечения);
- Тестер – составление плана тестирования и аттестации готового ПО (продукта), составление сценария тестирования, базовый пример, проведение мероприятий по плану тестирования;
- Разработчик – моделирование компонент программного обеспечения, кодирование.

Планирование проекта разработки программного обеспечения

Эффективное управление программным проектом напрямую зависит от правильного планирования работ, необходимых для его выполнения. План помогает руководителю предвидеть проблемы, которые могут возникнуть на каких-либо этапах создания ПО, и разработать превентивные меры для их предупреждения или решения. План,

разработанный на начальном этапе проекта, рассматривается всеми его участниками как руководящий документ, выполнение которого должно привести к успешному завершению проекта. Этот первоначальный план должен максимально подробно описывать все этапы реализации проекта.

Процесс планирования начинается, исходя из описания системы, с определения проектных ограничений (временные ограничения, возможности наличного персонала, бюджетные ограничения и т.д.). Эти ограничения должны определяться параллельно с оцениванием проектных параметров, таких как структура и размер проекта, а также распределением функций среди исполнителей. Затем определяются этапы разработки и то, какие результаты документация, прототипы, подсистемы или версии программного продукта) должны быть получены по окончании этих этапов. Далее начинается циклическая часть планирования. Сначала разрабатывается график работ по выполнению проекта или дается разрешение на продолжение использования ранее созданного графика. После этого проводится контроль выполнения работ и отмечаются расхождения между реальным и плановым ходом работ.

Далее, по мере поступления новой информации о ходе выполнения проекта, возможен пересмотр первоначальных оценок параметров проекта. Это, в свою очередь, может привести к изменению графика работ. Если в результате этих изменений нарушаются сроки завершения проекта, должны быть пересмотрены (и согласованы с заказчиком ПО) проектные ограничения.

Конечно, большинство руководителей проектов не думают, что реализация их проектов пройдет гладко, без всяких проблем. Желательно описать возможные проблемы еще до того, как они проявят себя в ходе выполнения проекта. Поэтому лучше составлять "пессимистические" графики работ, чем "оптимистические". Но, конечно, невозможно построить план, учитывающий все, в том числе случайные, проблемы и задержки выполнения проекта, поэтому и возникает необходимость периодического пересмотра проектных ограничений и этапов создания программного продукта.

План проекта должен четко показать ресурсы, необходимые для реализации проекта, разделение работ на этапы и временной график выполнения этих этапов. В некоторых организациях план проекта составляется как единый документ, содержащий все виды планов, описанных выше. В других случаях план проекта описывает только технологический процесс создания ПО. В таком плане обязательно присутствуют ссылки на планы других видов, но они разрабатываются отдельно от плана проекта.

Детализация планов проектов очень разнится в зависимости от типа разрабатываемого программного продукта и организации-разработчика. Но в любом случае большинство планов содержат следующие разделы.

1. Введение. Краткое описание целей проекта и проектных ограничений (бюджетных, временных и т.д.), которые важны для управления проектом.

2. Организация выполнения проекта. Описание способа подбора команды разработчиков и распределение обязанностей между членами команды.

3. Анализ рисков. Описание возможных проектных рисков, вероятности их проявления и стратегий, направленных на их уменьшение.

4. Аппаратные и программные ресурсы, необходимые для реализации проекта. Перечень аппаратных средств и программного обеспечения, необходимого для разработки программного продукта. Если аппаратные средства требуется закупать, приводится их стоимость совместно с графиком закупки и поставки.

5. Разбиение работ на этапы. Процесс реализации проекта разбивается на отдельные процессы, определяются этапы выполнения проекта, приводится описание результатов ("выходов") каждого этапа и контрольные отметки.

6. График работ. В этом графике отображаются зависимости между отдельными процессами (этапами) разработки ПО, оценки времени их выполнения и распределение членов команды разработчиков по отдельным этапам.

7. Механизмы мониторинга и контроля за ходом выполнения проекта. Описываются предоставляемые руководителем отчеты о ходе выполнения работ, сроки их предоставления, а также механизмы мониторинга всего проекта.

План должен регулярно пересматриваться в процессе реализации проекта. Одни части плана, например график работ, изменяются часто, другие более стабильны. Для внесения изменений в план требуется специальная организация документопотока, позволяющая отслеживать эти изменения.

Общие сведения о требованиях к информационным системам

Проблемы, которые приходится решать специалистам в процессе создания программного обеспечения, очень сложны. Природа этих проблем не всегда ясна, особенно если разрабатываемая программная система инновационная. В частности, трудно чётко описать те действия, которые должна выполнять система. Описание функциональных возможностей и ограничений, накладываемых на систему, называется требованиями к этой системе, а сам процесс формирования, анализа, документирования и проверки этих функциональных возможностей и ограничений – разработкой требований.

Требования подразделяются на пользовательские и системные. Пользовательские требования – это описание на естественном языке (плюс поясняющие диаграммы) функций, выполняемых системой, и ограничений, накладываемых на неё. Системные требования – это описание особенностей системы (архитектура системы, требования к параметрам оборудования и т.д.), необходимых для эффективной реализации требований пользователя.

Первые шаги по разработке требований к информационным системам - анализ осуществимости

Разработка требований — это процесс, включающий мероприятия, необходимые для создания и утверждения документа, содержащего спецификацию системных требований. Для новых программных систем процесс разработки требований должен начинаться с анализа осуществимости. Началом такого анализа является общее описание системы и ее назначения, а результатом анализа — отчет, в котором должна быть четкая рекомендация, продолжать или нет процесс разработки требований проектируемой системы. Другими словами, анализ осуществимости должен осветить следующие вопросы.

1. Отвечает ли система общим и бизнес-целям организации-заказчика и организации-разработчика?

2. Можно ли реализовать систему, используя существующие на данный момент технологии и не выходя за пределы заданной стоимости?

3. Можно ли объединить систему с другими системами, которые уже эксплуатируются?

Критическим является вопрос, будет ли система соответствовать целям организации. Если система не соответствует этим целям, она не представляет никакой ценности для организации. В то же время многие организации разрабатывают системы, не соответствующие их целям, либо не совсем ясно понимая эти цели, либо под влиянием политических или общественных факторов.

Выполнение анализа осуществимости включает сбор и анализ информации о будущей системе и написание соответствующего отчета. Сначала следует определить, какая именно информация необходима, чтобы ответить на поставленные выше вопросы. Например, эту информацию можно получить, ответив на следующее:

1. Что произойдет с организацией, если система не будет введена в эксплуатацию?

2. Какие текущие проблемы существуют в организации и как новая система поможет их решить?

3. Каким образом система будет способствовать целям бизнеса?

4. Требует ли разработка системы технологии, которая до этого не использовалась в организации?

Далее необходимо определить источники информации. Это могут быть менеджеры отделов, где система будет использоваться, разработчики программного обеспечения, знакомые с типом будущей системы, технологи, конечные пользователи и т.д.

После обработки собранной информации готовится отчет по анализу осуществимости создания системы. В нем должны быть даны рекомендации относительно продолжения разработки системы. Могут быть предложены изменения бюджета и графика работ по созданию системы или предъявлены более высокие требования к системе.

СИСТЕМНЫЙ ИНЖИНИРИНГ ПРОЕКТА ИНФОРМАЦИОННОЙ СИСТЕМЫ

Основные понятия методологии функционального моделирования IDEF0

IDEF0 (Integrated Definition Function Modeling) - методология функционального моделирования. В основе IDEF0 методологии лежит понятие блока, который отображает некоторую бизнес-функцию. Четыре стороны блока имеют разную роль: левая сторона имеет значение "входа", правая - "выхода", верхняя - "управления", нижняя - "механизма" (рисунок 2.1).

Взаимодействие между функциями в IDEF0 представляется в виде дуги, которая отображает поток данных или материалов, поступающий с выхода одной функции на вход другой. В зависимости от того, с какой стороной блока связан поток, его называют соответственно "входным", "выходным", "управляющим".

Рисунок 2.1 - Функциональный блок

Принципы моделирования в IDEF0

В IDEF0 реализованы три базовых принципа моделирования процессов:

принцип функциональной декомпозиции;

принцип ограничения сложности;

принцип контекста.

Принцип функциональной декомпозиции представляет собой способ моделирования типовой ситуации, когда любое действие, операция, функция могут быть разбиты (декомпозированы) на более простые действия, операции, функции. Другими словами, сложная бизнес-функция может быть представлена в виде совокупности элементарных функций. Представляя функции графически, в виде блоков, можно как бы заглянуть внутрь блока и детально рассмотреть ее структуру и состав (рисунок 2.2).

Принцип ограничения сложности. При работе с IDEF0 диаграммами существенным является условие их разборчивости и удобочитаемости. Суть принципа ограничения сложности состоит в том, что количество блоков на диаграмме должно быть не менее двух и не более шести. Практика показывает, что соблюдение этого принципа приводит к тому, что функциональные процессы, представленные в виде IDEF0 модели, хорошо структурированы, понятны и легко поддаются анализу.

Принцип контекстной диаграммы. Моделирование делового процесса начинается с построения контекстной диаграммы. На этой диаграмме отображается только один блок - главная бизнес-функция моделируемой системы. Если речь идет о моделировании целого предприятия или даже крупного подразделения, главная бизнес-функция не может быть сформулирована как, например, "продавать продукцию". Главная бизнес-функция системы - это "миссия" системы, ее значение в окружающем мире. Нельзя правильно сформулировать главную функцию предприятия, не имея представления о его стратегии.

При определении главной бизнес-функции необходимо всегда иметь ввиду цель моделирования и точку зрения на модель. Одно и то же предприятие может быть описано по-разному, в зависимости от того, с какой точки зрения его рассматривают: директор предприятия и налоговой инспектор видят организацию совершенно по-разному.

Контекстная диаграмма играет еще одну роль в функциональной модели. Она "фиксирует" границы моделируемой бизнес-системы, определяя то, как моделируемая система взаимодействует со своим окружением. Это достигается за счет описания дуг, соединенных с блоком, представляющим главную бизнес-функцию.

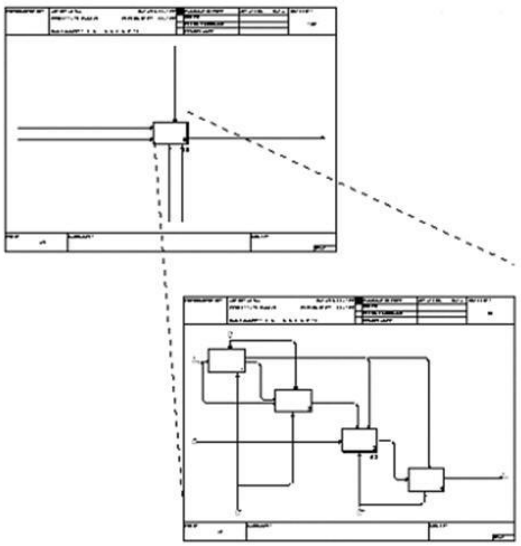

Рисунок 2.2 - Декомпозиция функционального блока

Пример

На рисунках 2.3, 2.4 представлен пример построения функциональной диаграммы, описывающей изготовление изделия. Рисунок 2.3 является примером построения контекстной диаграммы, рисунок 2.4 представляет собой первый уровень декомпозиции.

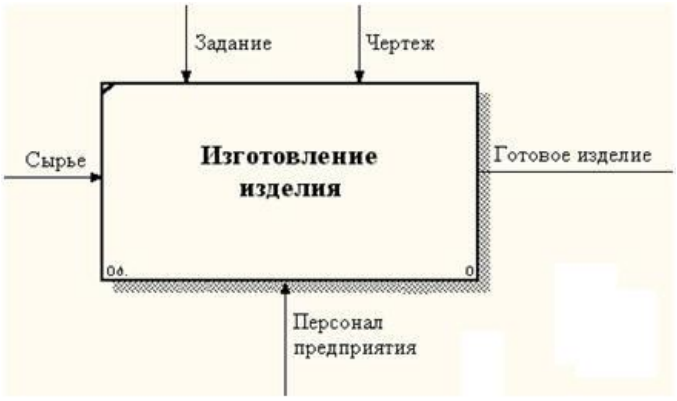

Рисунок 2.3 - Контекстная диаграмма

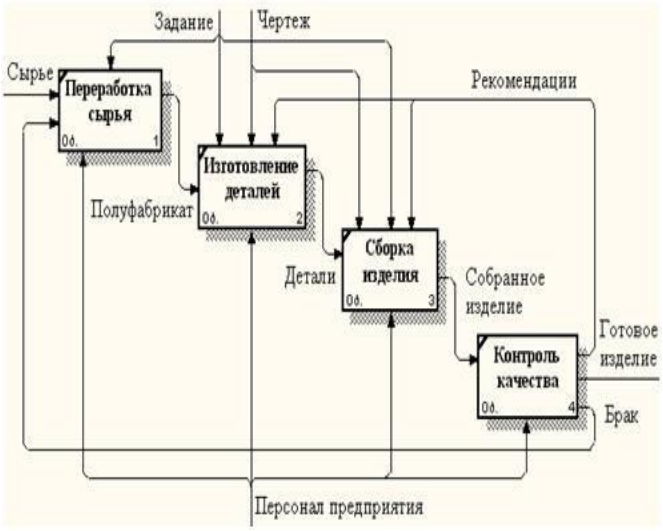

Рисунок 2.4 - Диаграмма первого уровня декомпозиции

Применение IDEF0

Существует два ключевых подхода к построению функциональной модели: построение “как есть” и построение “как будет”.

Построение модели “как есть”. Обследование предприятия является обязательной частью любого проекта создания или развития корпоративной информационной системы.

Построение функциональной модели “как есть” позволяет четко зафиксировать, какие деловые процессы осуществляются на предприятии, какие информационные объекты используются при выполнении деловых процессов и отдельных операций. Функциональная модель “как есть” является отправной точкой для анализа потребностей предприятия, выявления проблем и "узких" мест и разработки проекта совершенствования деловых процессов.

Построение модели “как будет”. Создание и внедрение корпоративной информационной системы приводит к изменению условий выполнения отдельных операций, структуры деловых процессов и предприятия в целом. Это приводит к необходимости изменения системы бизнес-правил, используемых на предприятии, модификации должностных инструкций сотрудников. Функциональная модель “как будет” позволяет уже на стадии проектирования будущей информационной системы определить эти изменения. Применение функциональной модели “как будет” позволяет не только сократить сроки внедрения информационной системы, но также снизить риски, связанные с невосприимчивостью персонала к информационным технологиям.

Метод описания процессов IDEF3

Для описания логики взаимодействия информационных потоков наиболее подходит IDEF3, называемая также workflow diagramming - методологией моделирования, использующая графическое описание информационных потоков, взаимоотношений между процессами обработки информации и объектов, являющихся частью этих процессов. Диаграммы Workflow могут быть использованы в моделировании бизнес-процессов для анализа завершенности процедур обработки информации. С их помощью можно описывать сценарии действий сотрудников организации, например последовательность обработки заказа или события, которые необходимо обработать за конечное время. Каждый сценарий сопровождается описанием процесса и может быть использован для документирования каждой функции.

IDEF3 - это метод, имеющий основной целью дать возможность аналитикам описать ситуацию, когда процессы выполняются в определенной последовательности, а также описать объекты, участвующие совместно в одном процессе,

Техника описания набора данных IDEF3 является частью структурного анализа. В отличие от некоторых методик описаний процессов IDEF3 не ограничивает аналитика

чрезмерно жесткими рамками синтаксиса, что может привести к созданию неполных или противоречивых моделей.

IDEF3 может быть также использован как метод создания процессов. IDEF3 дополняет IDEF0 и содержит все необходимое для построения моделей, которые в дальнейшем могут быть использованы для имитационного анализа.

Каждая работа в IDEF3 описывает какой-либо сценарий бизнес-процесса и может являться составляющей другой работы. Поскольку сценарий описывает цель и рамки модели, важно, чтобы работы именовались отглагольным существительным, обозначающим процесс действия, или фразой, содержащей такое существительное.

Точка зрения на модель должна быть задокументирована. Обычно это точка зрения человека, ответственного за работу в целом. Также необходимо задокументировать цель модели - те вопросы, на которые призвана ответить модель.

Диаграммы. Диаграмма является основной единицей описания в IDEF3.

Единицы работы - Unit of Work (UOW). UOW, также называемые работами (activity), являются центральными компонентами модели. В IDEF3 работы изображаются прямоугольниками с прямыми углами и имеют имя, выраженное отглагольным существительным, обозначающим процесс действия, одиночным или в составе фразы, и номер (идентификатор); другое имя существительное в составе той же фразы обычно отображает основной выход (результат) работы, например, "Изготовление изделия".

Связи. Связи показывают взаимоотношения работ. Все связи в IDEF3 однонаправлены и могут быть направлены куда угодно, но обычно диаграммы IDEF3 стараются построить так, чтобы связи были направлены слева направо. В IDEF3 различают три типа стрелок, изображающих связи, стиль которых устанавливается через меню Edit/Arrow Style:

Старшая (Precedence) - сплошная линия, связывающая единицы работ (UOW), Рисуется слева направо или сверху вниз. Показывает, что работа-источник должна закончиться прежде, чем работа-цель начнется.

Отношения (Relational Link) - пунктирная линия для изображения связей между единицами работ (UOW) а также между единицами работ и объектами ссылок.

Потоки объектов (Object Flow) - стрелка с двумя наконечниками, применяется для описания того факта, что объект используется в двух или более единицах работы, например, когда объект порождается в одной работе и используется в другой.

Старшая связь и поток объектов. Старшая связь показывает, что работа-источник заканчивается ранее, чем начинается работа-цель. Часто результатом работы-источника становится объект, необходимый для запуска работы-цели. В этом случае стрелку, обозначающую объект, изображают с двойным наконечником. Имя стрелки должно ясно идентифицировать отображаемый объект. Поток объектов имеет ту же семантику, что и старшая стрелка.

Перекрестки (Junction). Окончание одной работы может служить сигналом к началу нескольких работ, или же одна работа для своего запуска может ожидать окончания нескольких работ. Перекрестки используются для отображения логики взаимодействия стрелок при слиянии и разветвлении или для отображения множества событий, которые могут или должны быть завершены перед началом следующей работы. Различают перекрестки для слияния (Fan-in Junction) и разветвления (Fan-out Junction) стрелок. Перекресток не может использоваться одновременно для слияния и для разветвления. Для внесения перекрестка служит кнопка в палитре инструментов - добавить в диаграмму перекресток Junction. В диалоге Junction Type Editor необходимо указать тип перекрестка. Смысл каждого типа приведен в таблице 2.1.

Таблица 2.1 - Типы перекрестков

Обозначение	Наименование	Смысл в случае слияния стрелок	Смысл в случае разветвления стрелок

&	Asynchronous AND	Все предшествующие процессы должны быть завершены	Все следующие процессы должны быть запущены
&	Synchronous AND	Все предшествующие процессы завершены одновременно	Все следующие процессы запускаются одновременно
O	Asynchronous OR	Один или несколько предшествующих процессов должны быть завершены	Один или несколько следующих процессов должны быть запущены
O	Synchronous OR	Один или несколько предшествующих процессов завершены одновременно	Один или несколько следующих процессов запускаются одновременно
X	XOR (Exclusive OR)	Только один предшествующий процесс завершен	Только один следующий процесс запускается

В отличие от IDEF0 в IDEF3 стрелки могут сливаться и разветвляться только через перекрестки.

Декомпозиция работ. В IDEF3 декомпозиция используется для детализации работ. Методология IDEF3 позволяет декомпозировать работу многократно, т.е. работа может иметь множество дочерних работ. Это позволяет в одной модели описать альтернативные потоки. Возможность множественной декомпозиции предъявляет дополнительные требования к нумерации работ. Так, номер работы состоит из номера родительской работы, версии декомпозиции и собственного номера работы на текущей диаграмме (рисунок 2.5).

Рисунок 2.5 - Номер единицы работы (UOW)

5. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

5.1. Перечень основной и дополнительной литературы, необходимой для освоения дисциплины

5.1.1. Перечень основной литературы

1. Системная инженерия: учебное пособие/ Министерство образования и науки Российской Федерации, Федеральное государственное автономное образовательное учреждение высшего профессионального образования «Северо-Кавказский федеральный университет»; авт.-сост. В.И.Дроздова. - Ставрополь: СКФУ, 2014. - 115 с. - То же [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page= book&id =458082.

2. Маглинец Ю.А. Анализ требований к автоматизированным информационным системам [Электронный ресурс]/ Маглинец Ю.А.— Электрон. текстовые данные.— М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016.— 191 с.— Режим доступа: http://www.iprbookshop.ru/52184.— ЭБС «IPRbooks», по паролю

5.1.2. Перечень дополнительной литературы

1. Лиманова Н.И. Архитектура вычислительных систем и компьютерных сетей [Электронный ресурс]: учебное пособие/ Н.И. Лиманова. — Электрон. текстовые данные.

— Самара: Поволжский государственный университет телекоммуникаций и информатики, 2017. — 197 с. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru /75368.html.

2. Клименко И.С. Теория систем и системный анализ [Электронный ресурс]: учебное пособие / И.С. Клименко. — Электрон. текстовые данные. — М.: Российский новый университет, 2014. — 264 с. —Режим доступа: http://www.iprbookshop.ru/21322.html.

5.2. Перечень учебно-методического обеспечения самостоятельной работы обучающихся по дисциплине

1. Методические указания по выполнению лабораторных работ по дисциплине «Системная инженерия».

2. Методические рекомендации для студентов по организации самостоятельной работы по дисциплине «Системная инженерия».

5.3. Перечень ресурсов информационно-телекоммуникационной сети Интернет, необходимых для освоения дисциплины

1. Университетская библиотека online. http://www.biblioclub.ru.

2. ЭБС «IPRbooks». http://www.iprbookshop.ru.

3. Электронная библиотека СКФУ.. http://catalog.ncstu.ru.

4. Государственная публичная научно- техническая библиотека России. (ГПНТБ России). www.gpntb.ru.

Приложение А

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

на разработку ИС «Система»

Общие сведения

1.1. Наименование системы

Аналитическая информационная система «Система».

2.1. Назначение и цели создания системы

Система «Система» предназначена для информационного обеспечения процессов, которые происходят на кафедре связанных с учебно-методической, научной, общественной, организационно-методической и воспитательной работой.

Характеристика объектов информатизации

3.1. Краткое описание работы кафедры

К основным направлениям работы кафедры относятся:

- Учебно-методическая работа;
- Научная работа;
- Организационно-методическая работа;
- Работа со студентами заочниками;
- Общественная работа;
- Воспитательная работа.

…

3.2. Описание объектов информатизации

К основным объектам информатизации системы относятся:

Кафедра

- Наименование кафедры
- Факультет, к которому относится кафедра
- Веб-сайт кафедры
- Заведующий кафедрой

…

3.2.1. Учебно-методическая работа

План учебно-методической работы кафедры

- Учебный год
- Заведующий кафедрой, составивший план
- Кафедра

Тема для учебно-методической работы

- Названия работ
- Сроки исполнения
- Ответственные за выполнение темы

…

Требования к информационной системе

4.1. Базовые принципы разработки подсистем

При проектировании и разработке подсистем должны использоваться следующие базовые принципы:

- Исключение дублирования ввода информации и повышение ее достоверности, за счет отождествления ранее введенной информации;

…

Система должна удовлетворять следующим требованиям:

- Пользовательский интерфейс системы должен быть сформирован в соответствии с навыками и профилем пользователей;

…

Система должна содержать:

- Средства поиска информации;

…

Выбор прикладного программного обеспечения системы должен удовлетворять следующим критериям:

- Интеграция с базами данных, поддерживающих Web-технологии;

…

4.2. Требования к архитектуре системы.

Архитектура системы «Система» является трехзвенной. В качестве клиентского приложения выступает стандартный веб-браузер.

…

4.3. Требования к способам и средствам связи для информационного обмена между компонентами (модулями) Системы

Подсистемы должны взаимодействовать в пределах единой компьютерной сети (Интернет/Интранет), в которой происходит весь обмен информацией.

…

4.4. Требования к характеристикам взаимосвязей системы со смежными системами

Смежными системами для информационной системы «Система» являются: «Система2»,

…

4.5. Требования к режимам функционирования подсистемы

Разрабатываемая система должна функционировать 24 часа в сутки, 365 дней в году…

…

4.6. Требования к пользователям

Система подразумевает четыре типа пользователя:

- Сотрудник – имеет доступ к просмотру общих данных по своей кафедре, а также к просмотру и редактированию личных данных, имеет возможность ;

…

4.7. Требования по эргономике и технической эстетике

Основными требованиями по эргономике и технической эстетике является адекватность времени реакции модулей системы на сложность запроса пользователя к базам данных:

- При выполнении стандартных запросов пользователь должен работать с системой в реальном режиме времени;

…

4.8. Требования к численности и квалификации персонала системы и режиму его работы.

Квалификация персонала, порядок его подготовки и контроль знаний и навыков.

…

4.9. Требования к защите информации от несанкционированного доступа.

Разрабатываемая система должна обладать специализированной подсистемой разграничения доступа к информационным ресурсам, функционирующей на основе системы пользователей и пользовательских групп.

…

4.10. Требования к обмену данными

- Обмен данными должен происходить по сети в среде Intranet/Internet с поддержкой протокола TCP/IP;

…

4.11. Требования к внешней среде системы

Сервер баз данных или сервер приложений должен обеспечивать:

…

4.12. Требования к хранению данных

База данных «Система» должна содержать следующие данные:

- Данные о планировании учебно-методической работы;

…

4.13. Требования к отдельным подсистемам

4.13.1. Учебно-методическая работа

Функции заведующего кафедрой

- Создание плана учебно-методической работы на учебный семестр, заполнения, редактирования и удаления данных плана;

…

Состав и содержание работ по созданию Системы

Разработать модель БД, позволяющую хранить и обрабатывать все необходимые…

…

Приемо-сдаточные испытания Системы

После завершения всех работ по разработке компонентов, настройке подсистем и

…

Внесение корректировок в программный продукт, связанных с ошибками в Системе

Все ошибки, которые будут выявлены в работе Системы в течении 12 месяцев

…

Тестирование

Перед сдачей Модулей и Компонент Заказчику для выявления возможных сбоев в работе

…

Порядок контроля и приемки Системы

Для проверки выполнения заданных функций Системы, определения и проверки соответствия требованиям ТЗ количественных и (или) качественных характеристик Системы, выявления и устранения недостатков в действиях Системы и в разработанной документации, поэтапного контроля над ходом разработки должны быть проведены следующие виды испытаний:

- Предварительные;

…

Процедуры тестирования и контроля качества

При проведении испытаний должны использоваться следующие типы процедур тестирования и контроля качества:

- функциональное тестирование - тестирование ПО на соответствие функциональным спецификациям;

…

Общие требования к приемке работ

Сроки и место приемки, порядок приемки работ определяются в соответствии с настоящим ТЗ.

…

Требования к документированию

12.1. Требования к проектной документации

Состав и комплектность проектной документации должна соответствовать требованиям ГОСТ 34.201-89.

Перечень документации по созданию системы включает:

- Описание информационного обеспечения системы (П5);

…