Уфимский филиал Финуниверситета
Финансово-экономический колледж

Профессиональный модуль ПМ.02. МДК 02.01. «Разработка, внедрение и адаптация программного обеспечения отраслевой направленности»

Раздел 1. Тема 1.5., 1.6.

Преподаватель: Шершова Л.Н.

Уфа 2013

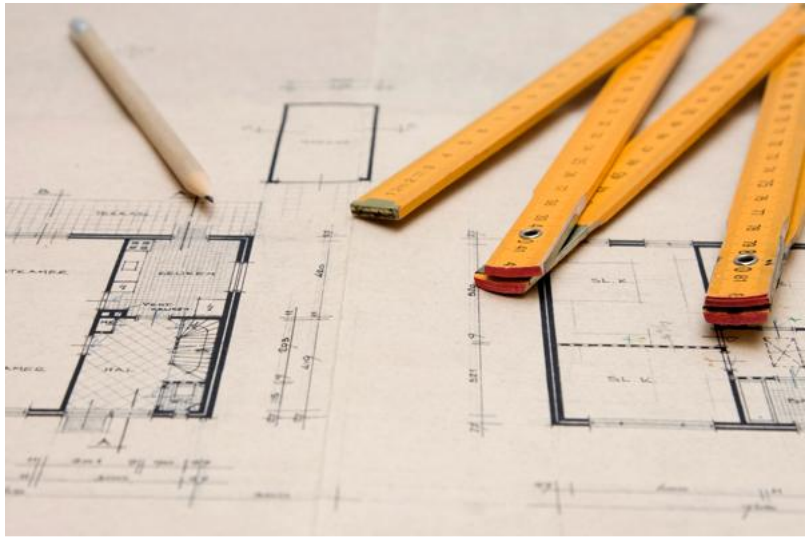
Раздел 1. Технология сбора информации для определения потребностей клиента

Тема 1.4. Особенности создания программного продукта

Принципы работы с требованиями к программному обеспечению. Проблематика проектирования

Согласно статистическим исследованиям группы Стендиша (Standish Group), в США ежегодно тратится более 250 млрд долларов на разработку приложений информационных технологий в рамках примерно 175 000 проектов. Причем 31 % проектов будет прекращен до завершения. Затраты на 52,7 % проектов составят 189 % от первоначальной оценки. В таком случае американские компании и правительственные учреждения потратят 81 млрд долларов на программные проекты, которые так и не будут завершены. Эти же организации заплатят дополнительно 59 млрд долларов за программные проекты, которые хотя и завершатся, но значительно превысят первоначально отведенное на них время .

Тема 1.4. Особенности создания программного продукта



Первым шагом на пути решения любой проблемы является осознание основных причин ее возникновения. В отчете группы Стендиша указано три наиболее часто встречающихся ключевых фактора, создающих проблемы при проектировании программного обеспечения:

- **недостаток исходной информации от клиента — 13 % всех проектов;**
- **неполные требования и спецификации — 12 % проектов;**
- **изменение требований и спецификаций — 12 % всех проектов.**

В остальном данные сильно расходятся. Конечно, проект может потерпеть неудачу из-за нереалистично составленного графика или неправильно распределенного времени (4 % проектов), нерационального подбора персонала и выделения ресурсов (6 %), несоответствия технологических навыков (7 %), а также по другим причинам. Тем не менее, если считать, что приведенные цифры представляют реальное положение дел в отрасли, то, по крайней мере, неудачи третьей части проектов объясняются причинами, непосредственно связанными со сбором и документированием требований, а также с управлением ими.

Тема 1.4. Особенности создания программного продукта

Несмотря на то что большинство проектов действительно превышает отведенное время и бюджет, оказалось, что около 9 % проектов крупных компаний были завершены вовремя и в пределах бюджета; аналогичного успеха удалось достигнуть в 16 % проектов мелких компаний. Возникает очевидный вопрос: «Каковы главные "факторы успеха" в этих проектах?» Согласно проведенному исследованию тремя наиболее важными факторами были следующие:

- **подключение к разработке пользователя — 16 % всех успешных проектов;**
- **поддержка со стороны исполнительного руководства — 14 % всех успешных проектов;**
- **четкая постановка требований — 12 % всех успешных проектов.**

Двумя самыми главными проблемами, упоминавшимися почти в половине ответов, оказались:

- **спецификации требований;**
- **управление требованиями клиента.**

Оценка стоимости ошибок

Некоторое время назад ряд компаний провел исследование оценки стоимости ошибок, возникающих на разных этапах создания программ. Каждая фирма действовала независимо, тем не менее результаты получены примерно одинаковые: если стоимость усилий, необходимых для обнаружения и устранения ошибок на стадии написания кода, принять за единицу, то стоимость выявления и устранения ошибки на стадии выработки требований будет в 5—10 раз меньше, а стоимость обнаружения и устранения ошибки на стадии сопровождения — в 20 раз больше. Откуда берется такая высокая стоимость ошибки? Ко времени обнаружения ошибки в требованиях группа разработчиков уже могла потратить время и усилия на создание проекта по этим ошибочным требованиям. В результате проект, вероятно, придется отбросить или пересмотреть.

Тема 1.4. Особенности создания программного продукта

Истинная природа ошибки может быть замаскирована; при проведении тестирования и проверок на данной стадии все думают, что имеют дело с ошибками проектирования, и значительное время и усилия могут быть потрачены впустую. В зависимости от того, где и когда при работе над проектом разработки программного приложения был обнаружен дефект, цена его может разниться в 50—100 раз. Причина состоит в том, что для его исправления придется затратить средства на некоторые (или все) нижеперечисленные действия.

1. Повторная спецификация.

2. Повторное проектирование.

3. Повторное кодирование.

4. Повторное тестирование.

5. Замена заказа — сообщить клиентам и операторам о необходимости заменить дефектную версию исправленной.

6. Внесение исправлений — выявить и устранить все неточности, вызванные неправильным функционированием ошибочно специфицированной системы, что может потребовать выплаты определенных сумм возмущенным клиентам, повторного выполнения определенных вычислительных задач на ЭВМ и т. п.

7. Списание той части работы (кода, части проектов и т. п.), которая выполнялась с наилучшими побуждениями, но оказалась ненужной, когда обнаружилось, что все это создавалось на основе неверных требований.

8. Отозвание дефектных версий встроенного программного обеспечения и соответствующих руководств. Если принять во внимание, что программное обеспечение сегодня встраивается в различные изделия — от наручных часов и микроволновых печей до автомобилей, — такая замена может коснуться как этих изделий, так и встроенного в них программного обеспечения.

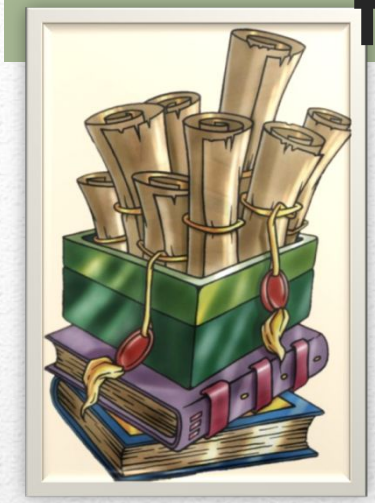
9. Выплаты по гарантийным обязательствам.

10. Ответственность за изделие, если клиент через суд требует возмещение убытка, причиненного некачественным программным продуктом.

11. Затраты на обслуживание представитель компании должен посетить клиента, чтобы установить новую версию программного обеспечения.

12. Создание документации.

Тема 1.5. Управление требованиями



Требования задают возможности, которые должна предоставлять система, так что соответствие или несоответствие некоторому множеству требований часто определяет успех или неудачу проекта. Поэтому имеет смысл узнать, что собой представляют требования, записать их, упорядочить и отслеживать их изменения. Определение управления требованиями выглядит следующим образом.

Управление требованиями — это систематический подход к выявлению, организации и документированию требований к системе, а также процесс, в ходе которого вырабатывается и обеспечивается соглашение между заказчиком и выполняющей проект группой по поводу меняющихся требований к системе. Учитывая, что системе будут предъявлены сотни, если не тысячи требований, то очень важно организовать их. Поскольку невозможно удерживать в памяти более нескольких десятков фактов, для успешного взаимодействия различных участников процесса необходимо обеспечить документирование требований. Требования следует записать так, чтобы они были доступны для ознакомления; это может быть документ, модель, база данных или листок на доске объявлений. Кроме того, очень важными факторами являются размер проекта и его сложность. Управление требованиями наиболее важно в больших проектах, в которых участвует множество людей и число требований к проекту велико. Допустим, таких требований 1000. Тогда придется столкнуться с задачами организации, определения приоритетов, управления доступом, а также обеспечения ресурсов для выполнения всех этих требований.

Тема 1.5. Управление требованиями

У пользователя есть технические или бизнес-задачи, для решения которых им нужны программисты. Задача последних состоит в том, чтобы понять проблемы пользователей в их собственной проблемной плоскости и на их языке и построить системы, удовлетворяющие их требованиям.

Программисты должны понять потребности пользователей и других заинтересованных лиц, на чью жизнь повлияет создание программы. Следующим шагом осуществляется переход в область решения — непосредственно к программированию. Однако для начала будет полезно сформулировать знания о предметной области. На данном этапе составляется список функций, которые должна реализовывать система.

Для того чтобы провести анализ, полезно определить, что же собственно представляет собой проблема. По определению Гауса и Вайнберга, проблема — это разница между желаемым и воспринимаемым. Иногда самым простым решением является изменение бизнес-процесса, а не создание новой системы. Как всегда, начинать следует с определения цели. Цель анализа состоит в том, чтобы добиться лучшего понимания решаемой проблемы до начала разработки. Для этого необходимо осуществить следующие пять этапов.

1. Достигнуть соглашения об определении проблемы.

2. Выделить основные причины — вопросы, стоящие за проблемой.

3. Выявить заинтересованных лиц и пользователей.

4. Определить границу системы решения.

5. Выявить ограничения, которые необходимо наложить на решение.

Тема 1.5. Управление требованиями

Преграды на пути выявления требований

Синдром «да, но...»

Одну из самых неприятных проблем, с которыми сталкиваются разработчики, можно назвать синдромом «да, но...». Это первичная наблюдаемая реакция пользователя на каждый

разработанный фрагмент программного обеспечения, которая могла бы быть выражена следующими словами: • «О, это действительно здорово! Можем реально использовать это, классная работа, молодцы, мальчики!» и т. д. • «Да, но как насчет?.. Нельзя ли было?.. А что, если?.. Причина синдрома «да, но...» кроется глубоко в природе проектирования программного обеспечения как интеллектуального неосязаемого процесса. Проблема усугубляется тем, что команда разработчиков крайне редко предоставляет что-либо пользователям для обсуждения до окончания разработки (создания программного кода). Реакция пользователей является следствием человеческой природы. Подобную реакцию можно часто наблюдать и при других повседневных обстоятельствах. Пользователи никогда ранее не видели новую систему или что-либо подобное; они не понимают, что программисты подразумевают, когда описывают ее. И вот теперь она перед ними — впервые после стольких месяцев (или лет) ожидания они имеют возможность взаимодействовать с системой. И оказывается, что это не совсем то, чего они ожидали! Как это ни грустно, но нужно принять факт существования синдрома «да, но...» в качестве объективной реальности и сделать некоторые выводы, которые помогут членам команды смягчить влияние этого синдрома в будущих проектах:

- синдром «да, но...» является следствием человеческой природы и неотъемлемой частью разработки любого приложения;
- разработчики могут существенно уменьшить воздействие этого синдрома путем применения методов, которые выявят эти «но» как можно раньше. Выявив их на более ранних этапах, можно направить большую часть усилий на разработку программ, которые уже прошли тест на «да, но...»

Тема 1.5. Управление требованиями

Преграды на пути выявления требований

Синдром «пользователь и разработчик»

Синдром «пользователь и разработчик» является следствием расхождения взглядов пользователей и разработчиков. Пользователи и разработчики, как правило, принадлежат к различным мирам, говорят на разных языках и имеют различный опыт, мотивацию и цели. Предлагаются рекомендации по смягчению данной ситуации.

Проблема	Решение
Пользователи не знают, чего хотят, а если и знают, то не могут это выразить	Признать пользователя экспертом в предметной области и ценить его в этом качестве; пытаться использовать альтернативные методы общения и выявления требований
Пользователи думают, что они знают, чего хотят, до тех пор, пока разработчики не предоставят им то, что те якобы хотели	Как можно раньше предлагать альтернативные методы выявления: раскадровку, ролевые игры, прототипы и т. п.
Аналитики думают, что они понимают проблемы пользователя лучше его самого	Поставить аналитика на место пользователя. Провести ролевую игру в течение часа или всего дня
Все считают, что другие руководствуются политическими мотивами	Такова человеческая натура, поэтому пусть все остается, как есть

Тема 1.5. Управление требованиями

Преграды на пути выявления требований

Функции

Использование функций — удобный способ описания возможностей без лишних подробностей. Такой подход имеет недостаток. Если команда при обсуждении не поймет, какая потребность стоит за функцией, это может привести к неприятным последствиям. Тем не менее это высокий уровень абстракции, удобный для описания возможностей системы. Рекомендуемое количество функций, которое дает полное представление о разрабатываемой системе, — 25—99, однако желательно, чтобы их число не превышало 50. После того как все функции перечислены, можно

приступить к принятию решения вида «отложить до следующей версии», «реализовать немедленно», «полностью отвергнуть» или «исследовать дополнительно». Это процесс корректировки масштаба лучше проводить на уровне функций, а не на уровне требований, иначе можно увязнуть в деталях. Чтобы лучше работать с этой информацией, введем понятие атрибутов функций — элементов данных, которые обеспечат дополнительную информацию о каждой функции.

Атрибут	Описание/примеры значений функции
Статус	• Предлагаемая • Утверждаемая • Включенная
Приоритет/полезность	• Критичная • Важная • Полезная
Трудоемкость	• Низкий уровень • Средний уровень • Высокий уровень
Риск	Вероятность того, что функция вызовет нежелательные последствия, такие как увеличение расходов, отставание от графика или даже закрытие проекта. • Высокий • Средний • Низкий
Стабильность	Вероятность того, что данная функция будет меняться или будет меняться ее понимание командой. • Высокая • Средняя • Низкая
Целевая версия	Указание версии продукта, в которой впервые появится реализация данной функции
Назначение	Информация для разработчиков
Обоснование	Ссылка на источник запрашиваемой функции

Преграды на пути выявления требований

Методы выявления требований

- интервьюирование и анкетирование;
- совещания посвященные требованиям;
- мозговой штурм и отбор идей;
- раскадровки;
- прецеденты;
- обыгрывание ролей;
- создание прототипов.

Контрольные вопросы и задания

ВОПРОСЫ:

1. Что такое спецификация, проект, кодирование?
2. Назовите 3 наиболее часто встречающихся фактора, создающих проблемы при проектировании.
3. Каковы главные «факторы успеха» в разработке проектов?
4. Как отличается стоимость ошибок, возникающих на разных этапах создания программ и почему?
5. В случае исправления ошибки, допущенной при работе над проектом, на какие действия может быть придется затратить средства?
6. Что задают требования?
7. Что такое управление требованиями?
8. Что такое проблема по определению Гауса и Вайнберга?
9. В чем состоит цель анализа и какие 5 этапов следует осуществить для понимания решаемой проблемы?
10. Что такое «синдром «да, но…?»
11. Какие выводы помогут смягчить влияние синдрома «да, но…» на разработчиков?
12. Как решить проблему «пользователь и разработчик»?
13. Что позволяют описывать функции?
14. Что такое атрибуты функций, назовите их?
15. Перечислите методы выявления требований.

ЗАДАНИЯ:

1. Опишите каждый из 5 этапов, которые необходимо осуществить для того, чтобы добиться лучшего понимания решаемой проблемы до начала разработки (см. Гагарина Л.Г. Технология разработки программного обеспечения, глава 2, п. 2.1.4, стр. 38-40).
2. Охарактеризуйте каждый метод выявления требований (см. Гагарина Л.Г. Технология разработки программного обеспечения, глава 2, п. 2.1.5, стр. 43-46).