Введение

Введение

В 60-х-70-х годах XX века вопрос о том, к какому роду деятельности относится программирование активно обсуждался на научных конференциях.

Дэвид Грайс утверждает, что написание программ — это наука (Gries, 1981). Дональд Кнут считает это искусством (Knuth, 1998). Уоттс Хамфри говорит, что это процесс (Humphrey, 1989), и т.д., существует ещё множество и других метафор о программировании.

Метафоры помогают понять процесс разработки ПО, сопоставляя его с другими, более знакомыми процессами.

Введение

В настоящий момент можно добавить к этим популярным трактовкам еще одну: «программирование - это бизнес, или производство». Чтобы понять, что программирование это бизнес, достаточно посмотреть, какими числами выражаются доходы современных IT-компаний.

Существует множество программ, которые задумываются, разрабатываются, сопровождаются и используются одним и тем же человеком. Обычно это начинающий программист или профессионал, работающий изолированно. Такие системы, как правило, имеют очень ограниченную область применения и короткое время жизни.

Введение

Существенная черта промышленной программы — уровень сложности: один разработчик практически не в состоянии охватить все аспекты такой системы. Т.е. сложность промышленных программ превышает возможности человеческого интеллекта - эта сложность здесь неизбежна: с ней можно справиться, но избавиться от нее нельзя. Сложность вызывается четырьмя основными причинами: сложностью реальной предметной области, из которой исходит заказ на разработку; трудностью управления процессом разработки; необходимостью обеспечить достаточную гибкость программы; неудовлетворительными способами описания поведения больших дискретных систем.

Введение

Сложность промышленных программ приводит к частым неудачам IT-проектов.

Причины неудачи IT-проектов.

Почему IT-проекты терпят неудачи: почему, казалось бы, хорошо спланированный проект не укладывается во временные рамки; почему по прошествии некоторого времени выясняется, что имеющегося бюджета недостаточно; почему полученный в итоге продукт не пользуется спросом?

Проблема сложна и многогранна. Трудно перечислить все возможные причины неудачи. Остановимся кратко на некоторых из них, представляющихся наиболее существенными.

Введение

- **Нереалистичные временные рамки** - Правильно оценить время, необходимое для выполнения проекта, - сложная задача, решение которой часто не под силу даже опытным менеджерам. Существуют специальные критерии, которые помогают принимать правильные решения, такие как учет времени в человеко-часах и т.д. Тем не менее, задача остается сложной, колоссальное значение в ней имеет грамотный учет рисков.
- **Недостаток количества исполнителей** - Иногда менеджер решает сэкономить, иногда в ходе разработки выясняется, что задача сложнее, чем казалось, - проблема недостатка рабочих рук, так или иначе, возникает достаточно часто.

Введение

- **Размытые границы проекта** - Одна из наиболее серьезных причин неудачи проекта - нечетко сформулированные цели, неоднократно меняющиеся в ходе разработки.
- **Недостаток средств** - Известны две крайности при планировании бюджета: чрезмерное раздувание (подход пессимиста) и чрезмерное уменьшение (подход оптимиста). Использование первого подхода чаще всего (если только ваш заказчик не совсем дилетант) приводит к тому, что ваша команда теряет проект. Второй подход часто применяется не только в силу оптимизма менеджмента, но и в рекламных целях, чтобы любой ценой выиграть проект.

Введение

- **Нехватка квалифицированных кадров** - одна из существенных проблем отрасли. Технологии развиваются с такой скоростью, что профессионалы вынуждены все время обновлять свои знания. Относительная новизна самой области IT, с одной стороны, становящееся повсеместным внедрение информационных технологий во все сферы человеческой деятельности, с другой, а, значит, все возрастающий спрос на специалистов ведут к существенной нехватке квалифицированных кадров..
- **Другие причины** - ....

Введение

Быстрое *увеличение сложности* и *размеров* современных комплексов программ при одновременном *повышении ответственности выполняемых функций* резко повысило требования со стороны пользователей к их качеству, надежности функционирования и безопасности применения.

Это также привело к принципиальному изменению методов в этой сфере: к переходу от технологии индивидуального программирования отдельных небольших программ к коллективному созданию крупных комплексов программ инженерными методами проектирования и разработки.

Основные определения

В литературе и нормативно-технических документах встречаются несколько определений **программного обеспечения (ПО)** :

- Программы, процедуры, правила и соответствующая документации системы обработки информации.
- Компьютерные программы, процедуры и, возможно, соответствующая документация и данные, относящиеся к функционированию компьютерной системы.
- Совокупность программ системы обработки информации и программных документов, необходимых для эксплуатации этих программ.
- …

Основные определения

Под **программной инженерией** будем понимать системный подход к анализу, проектированию, оценке, реализации, тестированию, обслуживанию и модернизации программного обеспечения. Программная инженерия занимается разработкой систематических моделей и надежных методов производства высококачественного программного обеспечения.

Основные определения

Вместо словосочетания «программное обеспечение» часто используют другое – «программный продукт» (ПП). Будем далее считать, что это одно и то же. Одно из главных свойств программного продукта - продаваемость.

Продаваемость - залог успеха бизнеса по разработке программного обеспечения. Если вы собираетесь что-то разработать, это должно быть востребовано на рынке. При возникновении такой необходимости возникает и вечный вопрос: приобрести уже готовый продукт или разработать собственный.

Основные определения

В рамках разработки ПО часто говорят о жизненном цикле программы или программного продукта.

Жизненный цикл — это развитие системы, продукта, услуги, проекта или других изготовленных человеком объектов, начиная со стадии разработки концепции и заканчивая прекращением применения.

Жизненный цикл программного обеспечения охватывает промежуток времени от возникновения необходимости в нем до завершения его эксплуатации.

Основные определения

Немного истории. Появление понятия жизненного цикла ПО было связано с кризисом программирования, который наметился в конце 60-х – начале 70-х годов прошлого века. Суть кризиса состояла в том, что программные проекты все чаще стали выходить из-под контроля: нарушались сроки, превышались запланированные объемы финансирования, результаты не соответствовали требуемым. Многие проекты вообще не доводились до завершения. Кроме того, оказалось, что недостаточно разработать программу, а надо ее еще сопровождать и этап сопровождения часто требует больше средств, чем разработка. Ситуация была вызвана ростом сложности проектов.

Основные определения

Масштабы ее нарастали. Необходимо было принимать меры для радикального усовершенствования принципов и методов разработки ПО с учетом его развития и сопровождения. Заговорили о том, что надо обратиться к опыту промышленного проектирования и производства, где был накоплен результат успешной разработки не менее сложных проектов.

Методологическую основу промышленной инженерии составляет понятие **жизненного цикла (ЖЦ) изделия (продукта) как совокупности** всех действий, которые надо выполнить на протяжении всей «жизни» изделия. Смысл жизненного цикла состоит во взаимосвязанности всех этих действий..

Основные определения

Например, вы решили построить садовый домик, имея некоторый опыт строительных работ. Ориентируясь на здравый смысл, вы купили материалы, инструменты и построили дом.

Через год он начал оседать одним углом – вы забыли его спроектировать и рассчитать фундамент. Через два года жена сказала, что лестница на мансарду должна быть не на веранде, где и так мало места, а с улицы.

Лестница у вас была сработана фундаментально, а с улицы надо вырубать и перекрывать козырьки – вы не предусмотрели модификацию конструкции.

Основные определения

Через три года у вас сгорела проводка, которая была «надежно» спрятана и вам пришлось снимать всю облицовку, которую вы «надежно» прибивали 100 мм гвоздями – вы не предусмотрели ремонтопригодность.

Если бы вы изначально рассматривали строительство с позиции жизненного цикла, то на этапах проектирования и строительства вы подумали бы о надежности, модифицируемости и ремонтопригодности.

.

Жизненный цикл ПО

Итак, жизненный цикл промышленного изделия:

- последовательность этапов (фаз, стадий): проектирования, изготовления образца, организация производства, серийное производство, эксплуатация, ремонт, вывод из эксплуатации;
- состоящих из технологических процессов, действий и операций.

Организация промышленного производства с позиции жизненного цикла позволяет рассматривать все его этапы во взаимосвязи, что ведет к сокращению сроков, стоимости и трудозатрат.

Жизненный цикл ПО

Впервые о жизненном цикле ПО заговорили в 1968 г. в Лондоне, где состоялась встреча 22-х руководителей проектов по разработке ПО.

На встрече анализировались проблемы и перспективы проектирования, разработки, распространения и поддержки программ.

Применяющиеся принципы и методы разработки ПО требовали постоянного усовершенствования. Именно на этой встрече была предложена концепция жизненного цикла ПО (SLC – Software Lifetime Cycle) как последовательности шагов-стадий, которые необходимо выполнить в процессе создания и эксплуатации ПО.

Жизненный цикл ПО

Вокруг этой концепции было много споров. В 1970 г. У.У. Ройс (W.W. Royce) произвел идентификацию нескольких стадий в типичном цикле и было высказано предположение, что контроль выполнения стадий приведет к повышению качества ПО и сокращению стоимости разработки.

Основным стандартом, регламентирующим концепцию ЖЦ ПС, стал стандарт ISO/IEC 12207 - Information Technology - Software Life Cycle Processes, разработанный в 1995 (в России он известен как ГОСТ 12207. Процессы жизненного цикла программных средств).

Стандарт ISO/IEC 12207 будет рассмотрен в последующих лекциях.

Требования к ПО

Обобщая определения, которые даются разными авторами и с учетом стандарта IEEE:

Требования - это свойства, которыми должно обладать ПО для адекватного определения функций, условий и ограничений выполнения ПО, а также объемов данных, технического обеспечения и среды функционирования.

Опыт индустрии информационных технологий показывает, что вопросы, связанные с управлением требованиями, оказывают критически-важное влияние на программные проекты, в определенной степени – на сам факт успешного завершения проектов.

Требования к ПО

Только систематическая работа с требованиями позволяет корректным образом обеспечить моделирование предметной области (задач реального мира) и формулирование необходимых приемочных тестов для того, чтобы убедиться в соответствии создаваемого ПО критериям, заданным реальными практическими потребностями.

Требования отражают потребности заказчиков, пользователей и разработчиков, заинтересованных в создании ПО. Заказчик и разработчик совместно проводят сбор требований, их анализ, пересмотр, определение необходимых ограничений и документирование.

Требования к ПО

Различают **требования** к **продукту** и к **процессу**, а также **функциональные**, **нефункциональные** и **системные** требования.

Требования к продукту и к процессу определяют условия функционирования и режимы работы ПО в операционной среде, ограничения на структуру и память компьютеров, на принципы взаимодействия программ и компьютеров и т.п.

Функциональные требования определяют назначение и функции системы, а **нефункциональные** - условия выполнения ПО и доступа к данным.

Требования к ПО

Системные требования описывают требования к программной системе, состоящей из взаимосвязанных программных и аппаратных подсистем и разных приложений.

Требования могут быть *количественные* (например, количество запросов в сек., средний показатель ошибок и т.п.). Значительная часть требований относится к атрибутам *качества*: безотказность, надежность и др., а также к *защите и безопасности* как ПО, так и данных.

Рассмотрим пример структурирования групп требований как требований к продукту, который предложен одним из классиков дисциплины управления требованиями Карлом Вигерсом (Рис.1).

Требования к ПО

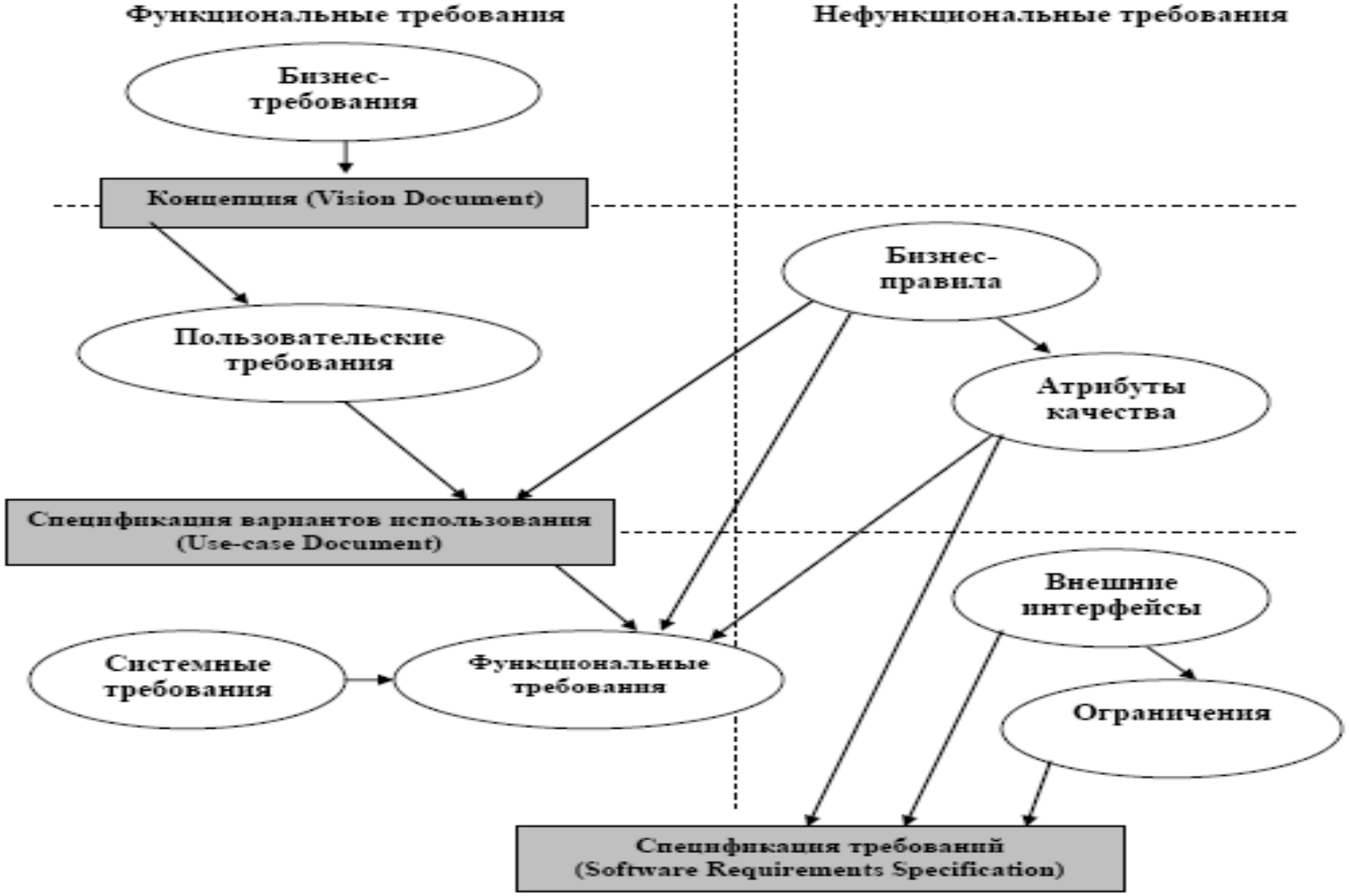

Рисунок 1 - Уровни требований по Вигерсу

Требования к ПО

Классификация различных категорий (видов) **требований** и связанных с ними понятий, важнейших с точки зрения их понимания и дальнейшего практического применения:

- **Потребности** (needs) - отражают проблемы бизнеса, персоналии или процесса, которые должны быть соотнесены с использованием или приобретением системы.

Группа функциональных требований:

- **Бизнес-требования** (Business Requirements) - определяют высокоуровневые цели организации или клиента (потребителя) - заказчика разрабатываемого программного обеспечения.

Требования к ПО

- **Пользовательские требования** (User Requirements) - описывают цели/задачи пользователей системы, которые должны достигаться/выполняться пользователями при помощи создаваемой программной системы. Эти требования часто представляют в виде вариантов использования (Use Cases).
- **Функциональные требования** (Functional Requirements) – определяют функциональность (поведение) программной системы, которая должна быть создана разработчиками для предоставления возможности выполнения пользователями своих обязанностей в рамках бизнес-требований и в контексте пользовательских требований.

Требования к ПО

Группа нефункциональных требований (Non-Functional Requirements):

- **Бизнес-правила** (Business Rules) - включают или связаны с корпоративными регламентами, политиками, стандартами, законодательными актами, внутрикорпоративными инициативами (например, стремление достичь зрелости процессов по некому стандарту), учетными практиками, алгоритмами вычислений и т.д.. На самом деле, достаточно часто можно видеть недостаточное внимание такого рода требованиям со стороны сотрудников ИТ-департаментов и, в частности, технических специалистов, вовлеченных в проект.

Требования к ПО

Business Rules Group дает понимание бизнес-правила, как «положения, которые определяют или ограничивают некоторые аспекты бизнеса. Они подразумевают организацию структуры бизнеса, контролируют или влияют на поведение бизнеса». Бизнес-правила часто определяют распределение ответственности в системе, отвечая на вопрос «кто будет осуществлять конкретный вариант, сценарий использования» или диктуют появление некоторых функциональных требований. В контексте дисциплины управления проектами (уже вне проекта разработки программного обеспечения, но выполнения бизнес-проектов и бизнес-процессов) такие правила обладают высокой значимостью и, именно они часто определяют ограничения бизнес-проектов, для автоматизации которых создается соответствующее программное обеспечение.

Требования к ПО

- **Внешние интерфейсы** (External Interfaces) - часто подменяются «пользовательским интерфейсом». На самом деле вопросы организации пользовательского интерфейса важны в данной категории требований, однако, конкретизация аспектов взаимодействия с другими системами, операционной средой (например, запись в журнал событий операционной системы), возможностями мониторинга при эксплуатации - все это не столько функциональные требования, сколько вопросы интерфейсов, так как функциональные требования связаны непосредственно с функциональностью системы, направленной на решение бизнес-потребностей.

Требования к ПО

- **Атрибуты качества** (Quality Attributes) - описывают дополнительные характеристики продукта в различных «измерениях», важных для пользователей и/или разработчиков. Атрибуты касаются вопросов портируемости, интероперабельности (прозрачности взаимодействия с другими системами), целостности, устойчивости и т.п..
- **Ограничения** (Constraints) - формулировки условий, модифицирующих требования или наборы требований, сужая выбор возможных решений по их реализации. В частности, к ним могут относиться параметры производительности, влияющие на выбор платформы реализации и/или развертывания (протоколы, серверы приложений, баз данных и т.п.).

Требования к ПО

- **Системные требования** (System Requirements) иногда классифицируются как составная часть группы функциональных требований (не путайте с как таковыми «функциональными требованиями»). Описывают высокоуровневые требования к программному обеспечению, содержащему несколько или много взаимосвязанных подсистем и приложений. При этом, система может быть как целиком программной, так и состоять из программной и аппаратной частей. В общем случае, частью системы может быть персонал, выполняющий определенные функции системы, например, авторизация выполнения определенных операций с использованием программно-аппаратных подсистем.

Требования к ПО

Необходимо сделать несколько важных замечаний по бизнес-правилам. Бизнес правила, как таковые, являются предметом пристального изучения различных специалистов в области как бизнес-моделирования, так и программной инженерии в целом.

Практика разработки программных требований включает идентификацию и описание бизнес-правил как самостоятельных артефактов (например, методология RUP выделяет отдельный артефакт Business Rule в рамках дисциплины Business Modeling).

Требования к ПО

Наравне с представленной классификацией требований, могут использоваться и другие подходы. Даже в рамках этой классификации, существуют и различные взгляды на ее интерпретацию и детализацию.

Например, как результат определения целевой аудитории и в рамках маркетинговой стратегии продвижения тиражируемого решения, возможно определять высокоуровневые возможности (ключевые характеристики, особенности) – «фичи» (features - признаки) разрабатываемого продукта.

Требования к ПО

Вигерс, описывает feature как «множество логически связанных функциональных требований, которые предоставляют определенные возможности для пользователя и удовлетворяют бизнес-целям <организации>»

С точки зрения требований, features являются самостоятельным артефактом, который может быть соотнесен как с функциональными требованиями, так и с нефункциональными (в т.ч. с ограничениями проектирования или атрибутами качества).

Отметим, что features обладают определенным дуализмом в своей интерпретации, зависимым от контекста конкретного продукта.

Требования к ПО

С одной стороны это может быть «тот самый список характеристик, указанный на коробке продукта» в случае создания «коробочного ПО», с другой стороны это может список высокоуровневых возможностей системы, например при заказной разработке ПО автоматизации бизнес-процессов конкретной организации.

Требования к ПО (Software Requirements)» состоит из следующих разделов:

- инженерия требований (Requirement Engineering);
- выявление требований (Requirement Elicitation);
- анализ требований (Requirement Analysis);
- спецификация требований (Requirement Specification);

Требования к ПО

- валидация требований (Requirement validation);
- управление требованиями (Requirement Management).

Инженерия требований к ПО - это дисциплина анализа и документирования требований к ПО, которая заключается в преобразовании предложенных заказчиком требований к системе в описании требований к ПО и их валидация. Она базируется на модели процесса определения требований в действующих лицах, обеспечивающих управление и формирование требований, а также на методах достижения показателей качества.

Требования к ПО

Модель процесса определения требований - это схема процессов ЖЦ, которые выполняются от начала проекта и до тех пор, пока не будут определены и согласованы требования. Данный процесс не является дискретным; это постоянно действующий процесс на всех этапах ЖЦ ПО. При этом процессом может быть маркетинг и проверка осуществимости требований в данном проекте.

Управление требованиями к ПО заключается в контроле за выполнением требований и планировании использования ресурсов (человеческих, программных, технических, временных, стоимостных) в процессе разработки промежуточных рабочих продуктов на этапах ЖЦ.

Требования к ПО

Качество и процесс улучшения требований - это процесс формулировки характеристик и атрибутов качества (надежность, реактивность и др.), которыми должно обладать ПО, методы их достижения на этапах ЖЦ и оценивания полученных результатов..

Выявление требований - это процесс извлечения информации из разных источников (договоров, материалов аналитиков по декомпозиции задач и функций системы и др.), проведения технических мероприятий (собеседований, собраний и др.) для формирования отдельных требований к продукту и к процессу разработки. Исполнитель должен согласовать требования с заказчиком.

Требования к ПО

Анализ требований – это процесс изучения потребностей и целей пользователей, классификация и преобразование их к требованиям к системе, аппаратуре и ПО, установление и разрешение конфликтов между требованиями, определение приоритетов, границ системы и принципов взаимодействия со средой функционирования.

Требования могут быть **функциональные** и **нефункциональные**, которые определяют соответственно внешние и внутренние характеристикам системы.

Требования к ПО

Спецификация требований к ПО - процесс формализованного описания функциональных и нефункциональных требований, требований к характеристикам качества, которые будут отрабатываться на этапах жизненного цикла ПО.

В спецификации требований отражается:

- структура ПО;
- требования к функциям, качеству и документации;
- задается в общих чертах архитектура системы и ПО, алгоритмы, логика управления и структура данных.

Требования к ПО

Специфицируются также системные требования, нефункциональные требования и требования к взаимодействию с другими компонентами и платформами (БД, СУБД, представление и хранение данных, сеть и др.).

Валидация требований - это проверка изложенных в спецификации требований, выполняющаяся для того, чтобы путем отслеживания источников требований убедиться, что они определяют именно данную систему. Заказчик и разработчик ПО проводят экспертизу созданного варианта требований с тем, чтобы разработчик мог далее проводить проектирование ПО.

Требования к ПО

Один из методов валидации - **прототипирование**, т.е. быстрая отработка отдельных требований на конкретном инструменте и исследование масштабов изменения требований, измерение объема функциональности и стоимости.

Верификация требований - это процесс проверки правильности спецификаций требований на их соответствие, непротиворечивость, полноту и выполнимость, а также на соответствие стандартам. В результате проверки требований делается согласованный выходной документ, устанавливающий полноту и корректность требований к ПО, а также возможность продолжить проектирование ПО.

Требования к ПО

Управление требованиями - это руководство процессами формирования требований на всех этапах ЖЦ и включает управление изменениями и атрибутами требований, а также проведение мониторинга - восстановления источника требований.

Управление изменениями возникает после того, когда ПО начинает работать в заданной среде и обнаруживаются ошибки в трактовке требований, либо в невыполнении некоторого отдельного требования и т.п.

.

Требования к ПО

Неотъемлемой составляющей процесса управления является трассировка требований для отслеживания правильности задания и реализации требований к системе и ПО на этапах ЖЦ, а также обратный процесс отслеживания в полученном продукте реализованных требований.

Для исправления некоторых требований или добавления нового требования составляется план изменения требований, который согласуется с заказчиком. Внесенные изменения влекут за собой и изменения в созданный продукт или в отдельные его компоненты..

Требования к ПО

Говоря о требованиях, нельзя не сказать об **участниках процессов** (Process Actors). Для них вводится понятие «роли» и дается понимание “ролей” для людей, которые участвуют в процессе работы с требованиями. Таких людей также называют «заинтересованными лицами» (software stakeholders).

Заинтересованное лицо - некто, имеющий возможность (в том числе, материальную) повлиять на реализацию проекта/продукта. .

Требования к ПО

Типичные примеры ролей:

- **Пользователи** (Users): группа, охватывающая тех людей, кто будет непосредственно использовать программное обеспечение; пользователи могут описать задачи, которые они решают (планируют решать) с использованием программной системы, а также ожидания по отношению к атрибутам качества, отображаемые в пользовательских требованиях.
- **Заказчики** (Customers): те, кто отвечают за заказ программного обеспечения или, в общем случае, являются целевой аудиторией на рынке программного обеспечения (образуют целевой рынок ПО);

Требования к ПО

- **Аналитики** (Market analysts): продукты массового рынка программного обеспечения (как и других массовых рынков, например, бытовой техники) не обладают «заказчиками» в понимании персонификации тех, кто «заказывает разработку». В то же самое время, лица, отвечающие за маркетинг, нуждаются в идентификации потребностей и обращению к тем, кто может играть роль т.н. <квалифицированных> «представителей» потребителей;
- **Регуляторы** (Regulators): многие области применения («домены») являются регулируемыми, например, телекоммуникации или банковский сектор.

Требования к ПО

Программное обеспечение для ряда целевых рынков (в первую очередь, корпоративного сектора) требует соответствия руководящим документам и прохождения процедур, определяемых уполномоченными органами.

- **Инженеры-программисты** (Software Engineer): лица, обладающие обоснованным интересом к разработке программного обеспечения, например, повторному использованию тех или иных компонент, библиотек, средств и инструментов. Именно инженеры ответственны за техническую оценку путей решения поставленных задач и последующую реализацию требований заказчиков.

Проектирование ПО

Проектирование ПО (Software design) - это процесс определения архитектуры, компонентов, интерфейсов, других характеристик системы и конечного состава программного продукта.

Целью проектирования является определение внутренних свойств программного обеспечения и детализации его внешних свойств, на основе выданных заказчиком и впоследствии проанализированных требований.

Проектирование ПО

В процессе создания программного обеспечения проектированию подлежит следующее:

- высокоуровневая архитектура программного обеспечения;
- низкоуровневая (внутренняя) архитектура программных компонентов;
- пользовательский интерфейс;
- сценарии взаимодействия пользователя с программным продуктом;
- структуры данных;
- модель предметной области;
- алгоритмы.

Проектирование ПО

На начальном этапе развития программной инженерии основными задачами проектирования были проектирование алгоритмов и структур данных. Помимо этого, проектировалась модульная структура программных приложений.

С появлением объектно-ориентированной парадигмы программирования проектируются взаимодействия между объектами классов, появляются объектно-ориентированные шаблоны микроархитектуры (шаблоны проектирования), например, такие как обозреватель, одиночка, фасад, прокси и другие. Более сложной организацией программного обеспечения является программная система, состоящая из компонентов.

Проектирование ПО

В настоящее время первоочередной задачей проектирования является проектирования архитектуры программных систем – макроархитектуры.

Проектирование интерфейса пользователя включает не только графический дизайн, но проектирование взаимодействия пользователя и программного приложения (User Experience design, UX design), где учитывается количество переходов и схемы между окнами, среднее время на выполнение определенного действия и другие параметры. Проектирование интерфейса пользователя существенно отличается для различных типов клиентских устройств и вида программного приложения (веб-приложение, полноэкранное игровое программное приложение, корпоративное приложение).

Проектирование ПО

Подход к проектированию программного приложения, при котором наибольшее значение имеет **качество человеко-машинного взаимодействия**, именуется Look and Feed Driven Design.

Данный подход имеет особую популярность для мобильных программных приложений, где эстетические предпочтения пользователя оказывают значительное влияние на решение о покупке и установке программного приложения.

Архитектура ПО

Упрощенное определение архитектуры – это разделение целой системы на части с учетом отношений между этими частями. Подобное разделение позволяет группе людей продуктивно работать и решать вместе намного больше задач, чем каждый из них смог их решить индивидуально.

Архитектура программного обеспечения – описание структуры программной системы, включающее программные компоненты, их свойства и отношения между ними.

Исходные сведения к процессу создания архитектуры ПО могут быть получены в виде ответов на следующие вопросы:.

Архитектура ПО

- Как пользователь будет использовать приложение?
- Каким способом ПО будет развертываться?
- Кем оно будет обслуживаться в процессе эксплуатации?
- Какие выдвинуты требования к качеству (безопасность, производительность, возможность параллельной обработки, интернационализация и конфигурация)?
- Каким образом спроектировать программное обеспечения с учетом требований гибкости и удобства обслуживания?
- Существуют ли обстоятельства, которые смогут влиять на архитектуру программного обеспечения сейчас или после его развертывания?

Архитектура ПО

Высокоуровневая архитектура программного обеспечения имеет следующие цели:

- раскрывать структуру системы, но скрывать детали реализации ее составных частей;
- реализовывать все варианты использования и сценарии;
- удовлетворять требованиям по функциональности и по качеству.

В процессе проектирования программной архитектуры важными являются следующие вопросы:

Архитектура ПО

- Какие части архитектуры являются фундаментальными, изменение которых в случае неверной реализации представляет наибольшие риски?
- Какие части архитектуры вероятнее всего подвергнуться изменениям, а также проектирование каких частей можно отложить?
- Какие условия могут привести к изменению спроектированной архитектуры?

Концептульная архитектура (conceptual architecture) – идейное представление об архитектуре программного приложения.

Архитектура ПО

Архитектурный стиль (reference architecture, шаблонная архитектура) – это абстрактная архитектура, применимая к некоторому множеству программных продуктов. Распространенными архитектурными стилями являются следующие архитектурные стили:

- клиент-серверный;
- многослойный;
- многоуровневый;
- сервис-ориентированный;
- шина сообщений.