Введение

В основе документа лежит показатель качества, также документ поделён на 6 частей. Каждая часть имеет свою тему в первой рассказывается основная цель показания качества и зачем оно нужно, во второй функциональные возможности , в третей рассказывается как удобно применять построение качества,в четвёртой рассказывается как сопровождать с продуктом, в пятой о мобильности , в шестой рассказывается о наиболее эффективном взаимодействии с по. Можно отметить , что затрагиваются очень важнее темы по написанию и сопровождению качества программы.

Основа

1

Вначале рассказывается введение для понимания основных процессов и понятий, а также основную задачу качества ПО.

2

Затем рассказывается функциональная часть , в которой описываются ГОСТ Р ИСО/МЭК 9126-93 и рассказывается о пригодности функционала атрибутов и требований ТЗ и ПС. Ещё идёт речь о важных местах в функционировании такие как завершённость, способы взаимодействия , точность, защита. Затем идет применение, в котором идёт речь о эргономичность, эффективность и привлекательность, в большинстве речь идёт о применении ПО к заказчику , затрагиваются темы по психологические и физиологические факторы. И важно ещё отметить что затрагивается теоретические основы для определения степени удобства и неудобства.

3

В следующем пункте рассказывается о сопровождаемости и поддержки ПО, и идет речь о специфических технических и,требований по управлению персоналом, занятым поддержкой ПО. Важно отметить, что затрагивается организации сопровождения уникальных работ, согласования с организационными целями, возможности сопровождения и многое другое.

4

Дальше идет речь о мобильность путём овита на вопросы "что?, зачем?,почему?",но это не главное главное в этой теме рассказ о свойствах,частях и работа механизмов ПО.

В заключении рассказывается об эффективности использовании методах перечисленных в предыдущих пунктах, и идёт разбор эффективности ПС,и о программных инструментах, ресурсов.

Вывод

В конце можно подвести итог, что данный текст рассказывает о важных темах качества по, и рассказываются о требованиях на ПС и связей нескольких частей путем тестирования.

Вопросы

1) **Что понимается под завершенностью программной системы?**

Завершенность ПС является наиболее общей характеристикой качества ПС для выражения ее функциональности. Завершенность как характеристика качества является мерой того, насколько требуемая спецификация реализована в данной системе.

В следствии возникновения в проектируемой ПС ошибок достигнутая при обеспечении его функциональности завершенность по факту может быть не такой, как ожидалось. Поэтому следует считать, что завершенность достигнута с некоторой вероятностью, которая определяется объемом и качеством проведенного системного и модульного тестирования. Повышение этой вероятность может быть произведено в случае продолжения тестирования с последующей отладкой ПС.

2) Какими основными аспектами характеризуется атрибут защищенности ПС?

Защищенность и безопасность функционирования представляют собой одну из наиболее трудно формализуемых характеристик сложных ПС, к числу которых нужно отнести следующие:

- соответствие установленным критериям и требованиям защиты от предумышленных и непредумышленных угроз безопасности ПС;
- соответствие методически верным методам и средствам защиты от проявления случайных ошибок в программах и данных;
- обеспечение эффективности оперативных методов защиты и восстановления при проявлениях угроз безопасности;
- соответствие нормативным документам и стандартам, регламентирующим способы защиты от различных типов угроз безопасности;
- обеспечение равнопрочности защиты и доступностью ресурсов для защиты.

К основным видам обеспечения безопасности ПС от искажения информационных и программных объектов относят защиты:

- от сбоев аппаратуры;
- от влияния «чужой» программы;
- от отказов «своей» программы;
- от ошибок пользователя;
- от несанкционированного доступа;
- от самой защиты

3) **Какие принципы позволяют обеспечить удобство использования ПО?**

Одним из ключевых показателей качества ПО, определяющим его эргономичность, эффективность и привлекательность, является удобство использования, или практичность. Оно определяется такими свойствами, как понятность пользовательского интерфейса, легкость обучения, трудоемкость решения функциональных задач, производительность работы пользователя, частота появления ошибок, сбоев и жалоб на неудобства при использовании.

В задачах проектирования удобного в использовании ПО используется большое количество информации, среди которой наиболее важные для разработки пользовательского интерфейса сведениями являются данные, характеризующие:

- свойственность человеку ошибаться;
- скоростные показатели деятельности пользователя при работе с программой и при выполнении должностных обязанностей;
- внимание человека; понятность программного интерфейса и принципов использования ПО;
- память человека;
- различие категорий пользователей;

Разработчикам интерфейсов необходимо учитывать как физические, так и умственные способности людей, которые будут использовать данное ПО. Для этого следует руководствоваться целым рядом принципов, позволяющих обеспечить удобство использования ПО.

Принцип учета знаний пользователя предполагает требование к интерфейсу быть настолько удобным в эксплуатации, чтобы пользователям не понадобились особые усилия для привыкания к нему. При этом следует использовать термины, понятные пользователю. Объекты, управляемые ПС, должны быть прямо связанные с рабочей деятельностью пользователя. Например, если создается система, предназначенная для авиадиспетчеров, то объектами управления в ней должны быть самолеты, взлеты и посадки, сигнальные знаки и т.п. Особенности системной реализацию интерфейса в терминах файловых структур и структур данных следует скрывать от пользователя.

Принцип согласованности интерфейса пользователя означает, что команды и меню системы должны иметь одинаковый формат, параметры следует передавать во все команды одинаково с сохранением похожей пунктуации команд. Такие интерфейсы позволяют существенно сократить время на обучение пользователей, причем навыки, полученные при изучении какой-то команды или отдельного элемента приложения, должны быть потом использованы при работе с другими частями ПС.

Принцип минимума неожиданностей предполагает, что количество неожиданностей должно быть как можно меньшим, так как пользователей раздражает, если система вдруг начинает себя вести непредсказуемо. Поскольку при работе с программной системой у пользователей формируется специфичная модель ее функционирования, то если действие пользователя в некоторой ситуации вызывает определенную реакцию системы, естественно ожидать, что такие же действие в другой ситуации приведут к аналогичной реакции. Поэтому разработчики пользовательских интерфейсов должны обеспечивать диалог так, чтобы похожие действия создавали похожий эффект.

Очень важным является принцип восстановления системы, потому что всегда имеется вероятность совершения пользователем ошибки. Грамотно и правильно разработанный интерфейс позволяет уменьшить количество ошибок пользователя.

Принцип поддержки пользователя, которая должна заключаться во включении в интерфейсе средств поддержки пользователей для обеспечения разных уровней помощи и необходимой справочной информацией.

Принцип учета разнородности пользователей предполагает, что с ПС могут работать разные типы пользователей. Разные группы пользователей по разному ее используют

4) **Дайте сравнительную оценку различных подходов к обеспечению мобильности ПС с целью их применения для разработки конкретной ПС?**

Необходимость в обеспечении мобильности возникает вследствие того, что быстрое развитие аппаратных средств компьютерной техники делает жизненный цикл многих больших программных систем намного продолжительнее, чем период оправданного эффективностью использования применения аппаратных комплексов, для которых они первоначально создавались.

Понятие мобильности включает в себя свойство кроссплатформенности.

Чтобы создавать переносимые программы, необходимо использовать следующие подходы.

1. Переиспользование бинарных файлов. Перенос приложения на новую программно-аппаратную платформу осуществляется без больших проблем для разработчиков, если старая и новая версии ПС совместимы на бинарном уровне, что означает поддержку новой системой двоичного интерфейса приложений ABI (Application Binary Interface).

Основными составляющими ABI являются:

- форматы исполняемых файлов и библиотек;
- набор библиотек и их функций, предоставляемых системой

2. Переиспользование исходного кода, что означает использование одного и того же исходного кода для сборки приложения на различных аппартнооперационных платформах. Такая возможность обеспечивается наличием использованием в целевых системах компиляторов для соответствующего языка программирования, а также наличием и доступность необходимых библиотек.
3. Использование интерпретируемого кода. Этот подход предполагает запись программного кода на интерпретируемых языках, применение которых не предполагает создания исполняемых файлов в формате целевой ОС, так как интерпретатор кода последовательно считывает и выполняет инструкции непосредственно из текста программы. Такая прямолинейная интерпретация однако является не эффективной, так как у интерпретатора практически отсутствуют возможности для оптимизации кода.
4. Использование эмуляторов ABI. ABI представляет собой набор соглашений для доступа приложения к ОС и другим низкоуровневым сервисам, спроектированный для переносимости исполняемого кода между машинами. В отличие от интерфейса прикладного программирования API (application programming interface), который регламентирует совместимость на уровне исходного кода, ABI можно рассматривать как набор правил, позволяющих компоновщику объединять откомпилированные модули без перекомпиляции всего кода. В ряде случаев сама ОС может обеспечивать бинарную совместимость с другой системой при помощи специального дополнительного слоя совместимости. Существует также продукты разных разработчиков, позволяющие загружать файлы других ОС посредством использования транслятора ABI, загружающего файлы требуемого формата, преобразуя вызовы функций и процедур, определенные внутри файла, в соответствующие вызовы текущей ОС
5. Виртуализация. Представляет собой альтернативу эмуляции ABI. Виртуализация предполагает запуск внутри основной ОС копии системы, которая с использованием программ эмулирует нужное аппаратное окружение. Такая программная эмуляция получила название виртуальной машины. На ней может быть установлена ОС и другое окружение, необходимое выполняемому приложению, хотя само это приложение запускается уже в родной для него среде.
6. Использование Web-технологий. Это наиболее современный подход, позволяющий строить распределенные системы, то есть системы, компоненты которых располагаются на разных компьютерах с разными апаратно-операционными платформами.

5) **Какими факторами определяется модифицируемость ПС?**

Модифицируемость программы определяется в основном свойствами документации, и свойствами, которые реализуются программным путем, она выражается через такие характеристики качества как расширяемость, модифицируемость, структурированность и модульность.

6) **Как понимается практически оправданный стиль программирования, чем он определяется?**

Практически оправданный стиль программирования - стиль программирования, который выбирают для рабочего процесса исходя из потребностей проекта (к примеру, расширяемости, читаемости) и его архитектуры.

7) **Каким образом использование Web-технологий позволяет обеспечить высокую мобильность ПС?**

Web-интерфейсы (браузеры или любая другая технология) присутствуют в любой популярной ОС. Поэтому, используя его, мы можем достигнуть максимальной кросплатфоменности с наименьшими затратами ресурсов.

8) **Как модульность ПС влияет на временную эффективность и эффективность по ресурсам?**

Любая модульная архитектура является более стабильной, но менее производительной, чем монолитная (хоть сейчас этот аспект постепенно сводится на нет), потому как появляются затраты и задержки в процессе доставки информации.