Управление программными проектами

###### 1. Менеджер программного проекта.

Выделенное лицо, взаимодействует 1) с заказчиком и планировщиком, и 2) распределяет работы между членами команды.

###### 2. Менеджмент программного проекта.

Управление проектом связано с вопросами планирования и организации работ, создания коллективов разработчиков и контроля за сроками и качеством выполняемых работ.

###### 7. Источники требований технического задания.

Группа инициаторов работ: заказчик (представитель); непосредственный пользователь (представитель); пользователь, применяющий результаты, полученные с помощью данной системы; инвестор; менеджер по продажам; покупатель.

###### 8. Проблемы оперирования требованиями.

Библиотекарь.

###### 9. Трассировка требований.

Трассировка требования – прохождение исходного требования через последовательность трансформаций от одного представления к другому, сопровождающееся соответствующим анализом.

Бывают: исходное представление, унифицированные представления, типизированное представление, модельные представления уровня анализа, модельные представления уровня конструирования, программные представления, документальные представления.

###### 10. Управление изменениями требований.

Требования: дополнительные, модифицирующие, отменяющие. Основной инструмент анализа – трассировка.

###### 11. Необходимость понятия жизненного цикла разработки.

ЖЦ ПО - это непрерывный процесс, который начинается с момента принятия решения о необходимости его создания и заканчивается в момент его полного изъятия из эксплуатации..

Стоимость ПО включает издержки в течение всего времени жизни системы, а не только затраты на разработку или исполнение программ.

###### 12. Чем определяется жизненный цикл разработки ПО.

Основным нормативным документом, регламентирующим ЖЦ ПО, является международный стандарт ISO/IEC 12207. Он определяет структуру ЖЦ, содержащую процессы, действия и задачи, которые должны быть выполнены во время создания ПО.

Структура ЖЦ ПО по стандарту ISO/IEC 12207 базируется на трех группах процессов:

* основные процессы ЖЦ ПО (приобретение, поставка, разработка, эксплуатация, сопровождение);
* вспомогательные процессы, обеспечивающие выполнение основных процессов (документирование, управление конфигурацией, обеспечение качества, верификация, аттестация, оценка, аудит, решение проблем);
* организационные процессы (управление проектами, создание инфраструктуры проекта, определение, оценка и улучшение самого ЖЦ, обучение).

###### 15. Внешние функции менеджера.

Создание условий для разработки, взаимодействие с заказчиком.

###### 16.Внутренние функции менеджера.

Распределение работ среди членов команды (управление и планирование).

###### 18. Ключевые роли коллектива разработчиков.

Архитектор проекта, проектировщики подсистем, руководители команд разработки подсистем, специалист по пользовательскому интерфейсу, эксперт предметной области.

###### 19. Априорное распределение кадровых ресурсов проекта.

* ориентировочный план требуемого кадрового состава, развернутого во времени.

Определяются кадровые потребности проекта, осуществляется привлечение сотрудников. Подбирается лидер коллектива.

###### 24. Жизненный цикл программного изделия.

Проекция пользовательского понятия «время жизни» на понятие разработчика «технологический цикл (цикл разработки)».

###### 25. Последовательное развитие проекта. Традиционная модель.

Предварительное техническое описание системы, которое строится в ходе специального этапа конструирования.

###### 26. Итеративное наращивание возможностей проекта.

На каждой фазе проекта строятся работоспособные продукты, развиваемые в дальнейшем путем обогащения функциональности и интерфейса. Недостающую работу можно будет выполнить на следующей итерации. Главная задача - как можно быстрее показать пользователям системы работоспособный продукт, тем самым активизируя процесс уточнения и дополнения требований.

###### 27. Объектно-ориентированная схема наращивания возможностей.

Итерации не отменяют результаты друг друга, но дополняют их.

###### 28. Возвратно-поступательная разработка.

Спиральная модель ЖЦ делает упор на начальные этапы ЖЦ: анализ и проектирование. На этих этапах реализуемость технических решений проверяется путем создания прототипов. Каждый виток спирали соответствует созданию фрагмента или версии ПО, на нем уточняются цели и характеристики проекта, определяется его качество и планируются работы следующего витка спирали. Таким образом углубляются и последовательно конкретизируются детали проекта и в результате выбирается обоснованный вариант, который доводится до реализации.

###### 29. Классическая модель.

Итерация – это исправление ошибок.

###### 30. Каскадная модель.

Вся разработка разбивается на этапы, причем переход с одного этапа на следующий происходит только после того, как будет полностью завершена работа на текущем. Каждый этап завершается выпуском полного комплекта документации, достаточной для того, чтобы разработка могла быть продолжена другой командой разработчиков.

Плюсы:

* на каждом этапе формируется законченный набор проектной документации, отвечающий критериям полноты и согласованности;
* выполняемые в логичной последовательности этапы работ позволяют планировать сроки завершения всех работ и соответствующие затраты.

Минусы:

* существенное запаздывание с получением результатов;
* согласование результатов с пользователями производится только в точках, планируемых после завершения каждого этапа работ, требования к ИС "заморожены" в виде технического задания на все время ее создания.

###### 42. CASE-технологии.

Под термином CASE-средства (Computer Aided Software Engineering, интегрированные технологии поддержки разработки ПО) понимаются программные средства, поддерживающие процессы создания и сопровождения ИС, включая анализ и формулировку требований, проектирование прикладного ПО (приложений) и баз данных, генерацию кода, тестирование, документирование, обеспечение качества, конфигурационное управление и управление проектом, а также другие процессы.

###### 43. Классификация средств CASE-технологий.

Обычно к CASE-средствам относят любое программное средство, автоматизирующее ту или иную совокупность процессов жизненного цикла ПО.

Классификация по типам:

* средства анализа (Upper CASE), предназначенные для построения и анализа моделей предметной области (Design/IDEF (Meta Software), BPwin (Logic Works));
* средства анализа и проектирования (Middle CASE), поддерживающие наиболее распространенные методологии проектирования и использующиеся для создания проектных спецификаций (Vantage Team Builder (Cayenne), Designer/2000 (ORACLE), Silverrun (CSA), PRO-IV (McDonnell Douglas), CASE.Аналитик (МакроПроджект)). Выходом таких средств являются спецификации компонентов и интерфейсов системы, архитектуры системы, алгоритмов и структур данных;
* средства проектирования баз данных, обеспечивающие моделирование данных и генерацию схем баз данных (как правило, на языке SQL) для наиболее распространенных СУБД. К ним относятся ERwin (Logic Works), S-Designor (SDP) и DataBase Designer (ORACLE). Средства проектирования баз данных имеются также в составе CASE-средств Vantage Team Builder, Designer/2000, Silverrun и PRO-IV;
* средства разработки приложений. К ним относятся средства 4GL (Uniface (Compuware), JAM (JYACC), PowerBuilder (Sybase), Developer/2000 (ORACLE), New Era (Informix), SQL Windows (Gupta), Delphi (Borland) и др.) и генераторы кодов, входящие в состав Vantage Team Builder, PRO-IV и частично - в Silverrun;
* средства реинжиниринга, обеспечивающие анализ программных кодов и схем баз данных и формирование на их основе различных моделей и проектных спецификаций. Средства анализа схем БД и формирования ERD входят в состав Vantage Team Builder, PRO-IV, Silverrun, Designer/2000, ERwin и S-Designor. В области анализа программных кодов наибольшее распространение получают объектно- ориентированные CASE-средства, обеспечивающие реинжиниринг программ на языке С++ (Rational Rose (Rational Software), Object Team (Cayenne)).

###### 44. Вспомогательные средства CASE-технологий.

Вспомогательные типы включают:

* средства планирования и управления проектом (SE Companion, Microsoft Project и др.);
* средства конфигурационного управления (PVCS (Intersolv));
* средства тестирования (Quality Works (Segue Software));
* средства документирования (SoDA (Rational Software)).

###### 45. Инструментальные средства CASE-технологий.

Инструментальное средство SE Companion является средой, в которой реализован электронный вариант методологии DATARUN.

###### 46. Интегрированные среды CASE-технологий.

Интегрированное CASE-средство (или комплекс средств, поддерживающих полный ЖЦ ПО) содержит следующие компоненты:

* репозиторий, являющийся основой CASE-средства. Он должен обеспечивать хранение версий проекта и его отдельных компонентов, синхронизацию поступления информации от различных разработчиков при групповой разработке, контроль метаданных на полноту и непротиворечивость;
* графические средства анализа и проектирования, обеспечивающие создание и редактирование иерархически связанных диаграмм (DFD, ERD и др.), образующих модели ИС;
* средства разработки приложений, включая языки 4GL и генераторы кодов;
* средства конфигурационного управления;
* средства документирования;
* средства тестирования;
* средства управления проектом;
* средства реинжиниринга.

###### 47. Управление временем проекта. Календарный план.

Календарный план – это поэтапно разбитая и упорядоченная по времени выполнения последовательность работ проекта.

Не нагляден при распараллеливании работ. Для этого используются сетевые графики.

###### 48. Управление рисками.

Риск – это любая причина, по которой развитие проекта может быть прервано. План управления рисками – идентификация рисков для данного проекта и мероприятия, снижающие зависимость проектов от рисков. Исключение риска, уменьшение риска, предупреждение ущерба от риска, планирование действий в непредвиденных ситуациях.

###### 52. Управление качеством. План управления качеством.

План управления качеством – это перечень мероприятий, которые проводятся в контрольных точках жизненного цикла проекта для измерения и оценки определенных показателей, характеризующих достигаемые результаты.