Проф. ФОМИН Н.С. (для ОМК)

ВОПРОСЫ К ЗАЧЕТУ С ОТВЕТАМИ ПО «ЖЦИС»

Вопрос 1. Понятие жизненного цикла ИС.

Жизненный цикл ИС— период времени, который начинается с момента принятия решения о необходимости создания информационной системы и заканчивается в момент ее полного изъятия из эксплуатации. Полный жизненный цикл информационной системы включает в себя, как правило, стратегическое планирование, анализ, проектирование, реализацию, внедрение и эксплуатацию.

Список использованных источников:

Баширова М.М. Учебное пособие по дисциплине «Управление жизненным циклом информационных систем» для направления 080500 «Бизнес-Информатика», профилей подготовки «Электронный бизнес» и «Архитектура» – Махачкала: ДГИНХ, 2013. – 114 с.

Вопрос 2. Жизненный цикл в методологии проектирования ИС.

Понятие жизненного цикла тесно связано с методологией проектирования информационных систем, т.к. методология проектирования информационных систем описывает процесс создания и сопровождения систем в виде жизненного цикла (ЖЦ) ИС, представляя его как некоторую последовательность стадий и выполняемых на них процессов. Для каждого этапа определяются состав и последовательность выполняемых работ, получаемые результаты, методы и средства, необходимые для выполнения работ, роли и ответственность участников, управленческие решения. Такое формальное описание ЖЦ ИС позволяет спланировать и организовать процесс коллективной разработки и обеспечить управление этим процессом.

Список использованных источников:

Баширова М.М. Учебное пособие по дисциплине «Управление жизненным циклом информационных систем» для направления 080500 «Бизнес-Информатика», профилей подготовки «Электронный бизнес» и «Архитектура» – Махачкала: ДГИНХ, 2013. – 114 с.

Вопрос 3. Основные стандарты жизненного цикла ИС.

Среди наиболее известных стандартов можно выделить следующие:

1. ГОСТ 34.601-90 - распространяется на ***автоматизированные системы*** и устанавливает стадии и этапы их создания. Кроме того, в стандарте содержится описание содержания работ на каждом этапе. Стадии и этапы работы, закрепленные в стандарте, в большей степени соответствуют каскадной модели жизненного цикла.
2. ISO/IEC 12207 (International Organization of Standardization /International ElectrotechnicalCommission)1995 – стандарт на процессы и организацию жизненного цикла. Распространяется на все виды заказного ПО. Стандарт не содержит описания фаз, стадий и этапов. Стандарт ISO/IEC 12207 определяет структуру жизненного цикла, включая процессы, действия и задачи, которые должны быть выполнены во время создания информационной системы. Согласно данному стандарту структура жизненного цикла основывается на трех группах процессов:
    - основные процессы жизненного цикла (приобретение, поставка, разработка, эксплуатация, сопровождение);
    - вспомогательные процессы, обеспечивающие выполнение основных процессов (документирование, ***управление конфигурацией***, обеспечение

качества, ***верификация***, ***аттестация***, оценка, ***аудит***, разрешение проблем);

- организационные процессы (управление проектами, создание инфраструктуры проекта, определение, оценка и улучшение самого жизненного цикла, обучение).

3. RationalUnifiedProcess (RUP) предлагает итеративную модель разработки, включающую четыре фазы: начало, исследование, построение и внедрение. Каждая фаза может быть разбита на этапы (итерации), в результате которых выпускается версия для внутреннего или внешнего использования. Прохождение через четыре основные фазы называется циклом разработки, каждый цикл завершается генерацией версии системы. Если после этого работа над проектом не прекращается, то полученный продукт продолжает развиваться и снова минует те же фазы. Суть работы в рамках RUP - это создание и сопровождение моделей на базе UML.8
4. MicrosoftSolutionFramework (MSF) сходна с RUP, так же включает четыре фазы: анализ, проектирование, разработка, стабилизация, является итерационной, предполагает использование ***объектно-ориентированного моделирования***. MSF в сравнении с RUP в большей степени ориентирована на разработку бизнес-приложений.
5. ExtremeProgramming (XP). Экстремальное программирование (самая новая среди рассматриваемых методологий) сформировалось в 1996 году. В основе методологии командная работа, эффективная коммуникация между заказчиком и исполнителем в течение всего проекта по разработке ИС, а разработка ведется с использованием последовательно дорабатываемых прототипов.

Вопрос 5. Что включает в себя полный жизненный цикл ИС?

Полный жизненный цикл информационной системы включает в себя, как правило:

- анализ;
- проектирование;
- разработка;
- тестирование;
- внедрение;
- сопровождение.
-

Вопрос 6. CASE-средство как организация процессов ЖЦ. Что такое CASE-средство? Какие процессы ЖЦ включает в себя CASE-средство?

Термин CASE (Computer Aided Software/System Engineering) используется в настоящее время в весьма широком смысле.

Первоначальное значение термина CASE ограничивалось лишь вопросами автоматизации разработки программного обеспечения. Однако в дальнейшем значение этого термина расширилось и приобрело новый смысл, охватывающий процесс разработки сложных информационных систем в целом.

Теперь под термином «CASE-средства» понимаются **программные средства, поддерживающие процессы создания и сопровождения информационных систем**, включая:

- анализ и формулировку требований;
- проектирование прикладного программного обеспечения и баз данных;
- генерацию кода;
- тестирование;
- документирование;
- обеспечение качества;
- конфигурационное управление и управление проектом;
- другие процессы.

Обычно к CASE-средствам относят любое программное средство, автоматизирующее ту или иную совокупность процессов жизненного цикла ПО и обладающее следующими основными характерными особенностями:

- мощные графические средства для описания и документирования ИС, обеспечивающие удобный интерфейс с разработчиком и развивающие его творческие возможности;
- интеграция отдельных компонент CASE-средств, обеспечивающая управляемость процессом разработки ИС;

- использование специальным образом организованного хранилища проектных метаданных (репозитория)

На сегодняшний день Российский рынок программного обеспечения располагает следующими наиболее развитыми CASE-средствами:

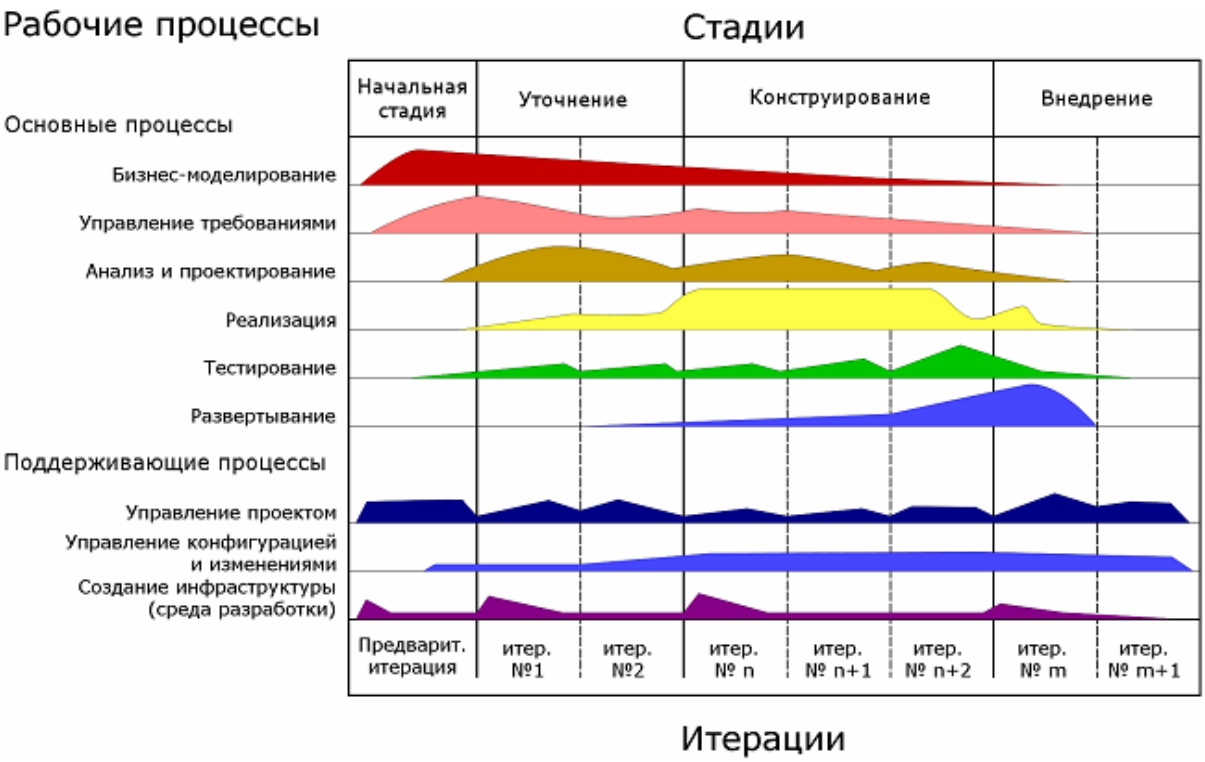
- Vantage Team Builder (Westmount I-CASE);
- Designer/2000;
- Silverrun;
- ERwin+BPwin;
- S-Designor;
- CASE.Аналитик.

Вопрос 7. Четыре стадии ЖЦ согласно методологии Rational Software.

Rational Software — компания-разработчик программного обеспечения. До 2003 года Rational была независимой компанией, в 2003 году её поглотила корпорация IBM.

Согласно методологии, предлагаемой Rational Software, жизненный цикл информационной системы подразделяется на четыре стадии:

- начало;
- уточнение;
- конструирование;
- передача в эксплуатацию.



Границы каждой стадии определены некоторыми моментами времени, в которые необходимо принимать определенные критические решения и, следовательно, достигать определенных ключевых целей.

Начальная стадия

На начальной стадии устанавливается область применения системы и определяются граничные условия. Для этого необходимо 11 идентифицировать все внешние объекты, с которыми должна взаимодействовать разрабатываемая система, и определить характер этого взаимодействия на высоком уровне. На начальной стадии идентифицируются все функциональные возможности системы и производится описание наиболее существенных из них.

На этой стадии работа ведется над:

- критерии успеха разработки;
- оценка риска;
- оценка ресурсов, необходимых для выполнения разработки;
- календарный план с указанием сроков завершения основных этапов.

Результатами начальной стадии являются:

- общее описание системы: основные требования к проекту, его характеристики и ограничения;

- начальная модель вариантов использования (степень готовности - 10-20%);
- начальный проектный глоссарий (словарь терминов);
- начальный бизнес-план;
- план проекта, отражающий стадии и итерации;
- один или несколько прототипов.

Стадия уточнения

На стадии уточнения проводится анализ прикладной области, разрабатывается архитектурная основа информационной системы. При принятии любых решений, касающихся архитектуры системы, необходимо принимать во внимание разрабатываемую систему в целом. Это означает, что необходимо описать большинство функциональных возможностей системы и учесть взаимосвязи между отдельными ее составляющими. В конце стадии уточнения проводится анализ архитектурных решений и способов устранения главных факторов риска в проекте.

Результатами стадии разработки являются:

- модель вариантов использования (завершенная по крайней мере на 80%), определяющая функциональные требования к системе;
- перечень дополнительных требований, включая требования нефункционального характера и требования, не связанные с конкретными вариантами использования;
- описание базовой архитектуры будущей системы;
- работающий прототип;
- уточненный бизнес-план;
- план разработки всего проекта, отражающий итерации и критерии оценки для каждой итерации.

Стадия конструирования

На стадии конструирования разрабатывается законченное изделие, готовое к передаче пользователю. По окончании этой стадии определяется работоспособность разработанного программного обеспечения.

Результатом стадии конструирования является продукт, готовый к передаче конечным пользователям. Как минимум, он содержит следующее:

- ПО, интегрированное на требуемых платформах;
- руководства пользователя;
- описание текущей реализации

Стадия передачи в эксплуатацию

На стадии передачи в эксплуатацию разработанное программное обеспечение передается пользователям. При эксплуатации разработанной системы в реальных условиях часто возникают различного рода проблемы, которые требуют дополнительных работ по внесению корректив в разработанный продукт. Это, как правило, связано с обнаружением ошибок и недоработок. В конце стадии передачи в эксплуатацию необходимо определить, достигнуты цели разработки или нет.

Вопрос 8. Понятие модели жизненного цикла.

Модель жизненного цикла информационной системы - структура процессов и действий, связанных с жизненным циклом, организуемых в стадии, которые также служат в качестве общей ссылки для установления связей и взаимопонимания сторон.

ГОСТ Р ИСО/МЭК 12207-2010. Информационная технология. Системная и программная инженерия. Процессы жизненного цикла программных средств — Введ. 2010-11-30. — М.: Стандартинформ, 2011. — 99 с.

Баширова М.М. Учебное пособие по дисциплине «Управление жизненным циклом информационных систем» для направления 080500 «Бизнес-Информатика», профилей подготовки «Электронный бизнес» и «Архитектура» – Махачкала: ДГИНХ, 2013. – 114 с.

Вопрос 9. Каскадная модель жизненного цикла. Понятие, особенности и основные этапы

Каскадная модель — модель процесса разработки программного обеспечения, жизненный цикл которой выглядит как поток, последовательно проходящий фазы анализа требований, проектирования. реализации, тестирования, интеграции и поддержки.

Особенность модели – переход на следующую ступень осуществляется только после того, как будет полностью завершена работа на предыдущей стадии; возвратов на пройденные стадии не предусматривается.

Основные этапы:

1. Анализ требований заказчика;

2. Проектирование;

3. Разработка;

4. Тестирование и опытная эксплуатация;

5. Сдача готового продукта.

На первом этапе проводится исследование проблемы, которая должна быть решена, четко формулируются все требования заказчика. Результатом, получаемым на данном этапе, является техническое задание (задание на разработку), согласованное со всеми заинтересованными сторонами.

На втором этапе разрабатываются проектные решения, удовлетворяющие всем требованиями, сформулированным в техническом задании. Результатом данного этапа является комплект проектной документации, содержащей все необходимые данные для реализации проекта.

Третий этап – реализация проекта. Здесь осуществляется разработка программного обеспечения (кодирование) в соответствии с проектными решениями, полученными на предыдущем этапе. Методы, используемые для реализации, не имеют принципиального значения. Результатом выполнения данного этапа является готовый программный продукт.

На четвертом этапе проводится проверка полученного программного обеспечения на предмет соответствия требованиям, заявленным в техническом задании. Опытная эксплуатация позволяет выявить различного рода скрытые недостатки, проявляющиеся в реальных условиях работы информационной системы.

Последний этап – сдача готового проекта. Главная задача этого этапа – убедить заказчика, что все его требования выполнены в полной мере.

Вопрос 10. Основные достоинства каскадной модели.

1. Стабильность требований в течение всего жизненного цикла разработки;

2. На каждом этапе формируется законченный набор проектной документации, отвечающий критериям полноты и согласованности. На заключительных этапах также разрабатывается пользовательская документация, охватывающая все предусмотренные стандартами виды обеспечения информационной системы (организационное, методическое, информационное, программное, аппаратное).

3. Выполняемые в логичной последовательности этапы работ позволяют планировать сроки завершения и соответствующие затраты.

4. Определенность и понятность шагов модели и простота её применения;

Вопрос 11. Основные недостатки каскадной модели.

1. Существенная задержка в получении результатов;

Данный недостаток проявляется в основном в том, что вследствие последовательного подхода к разработке согласование результатов с заинтересованными сторонами производится только после завершения очередного этапа работ. Может оказаться, что разрабатываемая информационная система не соответствует требованиям пользователей, причем такие несоответствия могут возникать на любом этапе разработки – искажения могут непреднамеренно вноситься и проектировщиками-аналитиками, и программистами, так как они не обязательно хорошо разбираются в тех предметных областях, для которых производится разработка информационной системы.

Кроме того, используемые при разработке информационной системы модели автоматизируемого объекта, отвечающие критериям внутренней согласованности и полноты, могут в силу различных причин устареть за время разработки (например, из-за внесения изменений в законодательство, колебания курса валют и т. п.). Это относится и функциональной модели, и к информационной модели, и к проектам интерфейса пользователя, и к пользовательской документации.

2. Ошибки и недоработки на любом из этапов проявляются, как правило, на последующих этапах работ, что приводит к необходимости возврата назад;

Данный недостаток каскадной модели является одним из проявлений предыдущего. Поэтапная и последовательная работа над проектом может быть следствием того, что ошибки, допущенные на более ранних этапах, как правило, обнаруживаются только на последующих стадиях работы над проектом. Поэтому, после того как ошибки проявятся, проект возвращается на предыдущий этап, перерабатывается и снова передается на последующую стадию. Это может служить причиной срыва графика работ и усложнения взаимоотношений между группами разработчиков, выполняющих отдельные этапы работы. Самым же неприятным является то, что недоработки предыдущего этапа могут обнаруживаться не сразу на последующем этапе, а позднее (например, на стадии опытной эксплуатации могут проявиться ошибки в описании предметной области). Это означает, что часть проекта должна быть возвращена на начальный этап работы. Вообще, работа может быть возвращена с любого этапа на любой предыдущий этап,

Одной из причин данной ситуации является то, что в качестве экспертов, участвующих в описании предметной области, часто выступают будущие пользователи системы, которые иногда не могут четко сформулировать то, что они хотели бы получить. Кроме того, заказчики и исполнители часто неправильно понимают друг друга вследствие того, что исполнители обычно не являются специалистами в предметной области решаемой задачи, а заказчики далеки от программирования.

3. Сложность параллельного ведения работ по проекту;

Отмеченные проблемы возникают вследствие того, что работа над проектом строится в виде цепочки последовательных шагов. Причем даже в том случае, когда разработку некоторых частей проекта (подсистем) можно вести параллельно, при использовании каскадной схемы распараллеливание работ весьма затруднительно. Сложности параллельного ведения работ связаны с необходимостью постоянного согласования различных частей проекта. Чем сильнее взаимозависимость отдельных частей проекта, тем чаще и тщательнее должна выполняться синхронизация, тем сильнее зависят друг от друга группы разработчиков. Поэтому преимущества параллельного ведения работ просто теряются.

Отсутствие параллелизма негативно сказывается и на организации работы всего коллектива разработчиков. Работа одних групп сдерживается другими. Пока производится анализ предметной области, проектировщики, разработчики и те, кто занимается тестированием и администрированием, почти не загружены. Кроме того, при последовательной разработке крайне сложно внести изменения в проект после завершения этапа и передачи проекта на следующую стадию. Так, например, если после передачи проекта на следующий этап группа разработчиков нашла более эффективное решение, оно не может быть использовано. Это связано с тем, что более раннее решение уже, возможно, реализовано и связано с другими частями проекта. Поэтому исключается (или, по крайней мере, существенно затрудняется) доработка проекта после его передачи на следующий этап.

4. Чрезмерная информационная перенасыщенность каждого из этапов;

Проблема информационной перенасыщенности возникает вследствие сильной зависимости между различными группами разработчиков. Данная проблема заключается в том, что при внесении изменений в одну из частей проекта необходимо оповещать всех разработчиков, которые использовали или могли бы использовать эту часть в своей работе. Когда система состоит из большого количества взаимосвязанных подсистем, то синхронизация внутренней документации становится важной самостоятельной задачей. При этом синхронизация документации на каждую часть системы – не более чем процесс оповещения групп разработчиков. Самим же разработчикам необходимо ознакомиться с изменениями и оценить, не сказались ли эти изменения на уже полученных результатах. Все это может требовать проведения повторного тестирования и даже внесения изменений в уже готовые части проекта. Причем эти изменения, в свою очередь, должны быть отражены во внутренней документации и разосланы другим группам разработчиков. Как следствие, объем документации по мере разработки проекта растет очень быстро, так что требуется все больше времени для составления документации и ознакомления с ней.

Следует также отметить, что, помимо изучения нового материала, не отпадает необходимость в изучении старой информации. Это связано с тем, что вполне вероятна ситуация, когда в процессе разработки изменяется

состав группы разработчиков (этот процесс носит название ротации кадров). Новым разработчикам необходима информация о том, что было сделано до них. Причем чем сложнее проект, тем больше времени требуется, чтобы ввести нового разработчика в курс дела.

5. Сложность управления проектом;

В основном сложность обусловлена строгой последовательностью стадий разработки и наличием сложных взаимосвязей между различными частями проекта. Последовательность разработки проекта приводит к тому, что одни группы разработчиков должны ожидать результатов работы других команд. Поэтому требуется административное вмешательство для согласования сроков работы и состава передаваемой документации.

В случае же обнаружения ошибок в выполненной работе необходим возврат к предыдущим этапам выполнения проекта. Это приводит к дополнительным сложностям в управлении проектом. Разработчики, допустившие просчет или ошибку, вынуждены прервать текущую работу (над новым проектом) и заняться исправлением ошибок. Следствием этого обычно является срыв сроков выполнения как исправляемого, так и нового проектов. Требовать же от команды разработчиков ожидания окончания следующей стадии разработки нерационально, так как приводит к существенным потерям рабочего времени.

Упростить взаимодействие между группами разработчиков и уменьшить информационную перенасыщенность документации можно, сокращая количество связей между отдельными частями проекта. Однако это обычно весьма непросто. Далеко не каждую информационную систему можно разделить на несколько слабо связанных подсистем.

6. Высокий уровень риска и ненадежность инвестиций.

Чем сложнее проект, тем больше продолжительность каждого из этапов разработки и тем сложнее взаимосвязи между отдельными частями проекта, количество которых также увеличивается. Причем результаты разработки можно реально увидеть и оценить лишь на этапе тестирования, то есть после завершения анализа, проектирования и разработки – этапов, выполнение которых требует значительного времени и средств. Как уже было отмечено, запоздалая оценка порождает серьезные проблемы при выявлении ошибок анализа и проектирования – требуется возврат проекта на предыдущие стадии и повторение процесса разработки. Однако возврат на предыдущие стадии может быть связан не только с ошибками, но и с изменениями, произошедшими в предметной области или в требованиях заказчика за время разработки. Причем возврат проекта на доработку вследствие этих причин не гарантирует, что предметная область снова не изменится к тому моменту, когда будет готова следующая версия проекта. Фактически это означает, что существует вероятность того, что процесс разработки «зациклится», и система никогда не дойдет до сдачи в

эксплуатацию. Расходы на проект будут постоянно расти, а сроки сдачи готового продукта постоянно откладываться.

Вопрос 12. Спиральная модель жизненного цикла. Понятие, особенности и основные этапы.

Спиральная модель — модель процесса разработки программного обеспечения, жизненный цикл которой выглядит как спираль, на каждом витке которой выполняется создание очередной версии продукта, уточняются требования проекта, определяется его качество и планируются работы следующего витка. Особое внимание уделяется начальным этапам разработки — анализу и проектированию, где реализуемость тех или иных технических решений проверяется и обосновывается посредством создания прототипов.

На этапах анализа и проектирования реализуемость технических решений и степень удовлетворения потребностей заказчика проверяется путем создания прототипов. Каждый виток спирали соответствует созданию работоспособного фрагмента или версии системы. Это позволяет уточнить требования, цели и характеристики проекта, определить качество разработки, спланировать работы следующего витка спирали. Таким образом углубляются и последовательно конкретизируются детали проекта и в результате выбирается обоснованный вариант, который удовлетворяет действительным требованиям заказчика и доводится до реализации.

Основные этапы:

1. Планирование – определение целей, вариантов и ограничений.

2. Анализ риска – анализ вариантов и распознавание/выбор риска.

3. Конструирование – разработка продукта следующего уровня.

4. Оценивание – оценка заказчиком текущих результатов конструирования.

Вопрос 13. Основные достоинства спиральной модели

1. Итерационная разработка существенно упрощает внесение изменений в проект при изменении требований заказчика.

2. При использовании спиральной модели отдельные элементы информационной системы интегрируются в единое целое постепенно. При итерационном подходе интеграция производится фактически непрерывно. Поскольку интеграция начинается с меньшего количества элементов, то возникает гораздо меньше проблем при ее проведении (по некоторым оценкам, при использовании каскадной модели разработки интеграция занимает до 40 % всех затрат в конце проекта).

3. Уменьшение уровня рисков. Данное преимущество является следствием предыдущего, так как риски обнаруживаются именно во время интеграции. Поэтому уровень рисков максимален в начале разработки проекта. По мере продвижения разработки ожидаемый уровень рисков снижается. Данное утверждение справедливо при любой модели разработки, однако при использовании спиральной модели снижение уровня рисков происходит с наибольшей скоростью. Это связано с тем, что при итерационном подходе интеграция выполняется уже на первой итерации, и на начальных итерациях выявляются многие аспекты проекта, такие как пригодность используемых инструментальных средств и программного обеспечения, квалификация разработчиков и т. п.

4. Итерационная разработка обеспечивает большую гибкость в управлении проектом, давая возможность внесения тактических изменений в разрабатываемое изделие. Например, можно сократить сроки разработки за счет снижения функциональности системы или использовать в качестве составных частей системы продукцию сторонних фирм вместо собственных разработок. Это может быть актуальным в условиях конкурентной борьбы, когда необходимо противостоять продвижению изделия, предлагаемого конкурентами.

5. Итерационный подход упрощает повторное использование компонентов (реализует компонентный подход к программированию – более подробно об этом мы будем говорить в следующей главе). Это обусловлено тем, что гораздо проще выявить (идентифицировать) общие части проекта, когда они уже частично разработаны, чем пытаться выделить их в самом начале проекта. Анализ проекта после проведения нескольких начальных итераций позволяет выявить общие многократно используемые компоненты, которые на последующих итерациях будут совершенствоваться.

6. Спиральная модель позволяет получить более надежную и устойчивую систему. Это связано с тем, что по мере развития системы ошибки и слабые места обнаруживаются и исправляются на каждой итерации. Одновременно могут корректироваться критические параметры эффективности, что в случае каскадной модели доступно только перед внедрением системы.

7. Итерационный подход дает возможность совершенствовать процесс разработки – анализ, проводимый в конце каждой итерации, позволяет проводить оценку того, что должно быть изменено в организации разработки, и улучшить ее на следующей итерации.

Вопрос 14. Основные недостатки спиральной модели.

1. Если проект имеет низкую степень риска или небольшие размеры, модель может оказаться дорогостоящей. Оценка рисков после прохождения каждой спирали связана с большими затратами;

2. Жизненный цикл модели имеет усложненную структуру, поэтому может быть затруднено её применение разработчиками, менеджерами и заказчиками;

3. Спираль может продолжаться до бесконечности, поскольку каждая ответная реакция заказчика на созданную версию может порождать новый цикл, что отдаляет окончание работы над проектом;

4. Большое количество промежуточных циклов может привести к необходимости в обработке дополнительной документации;

5. Использование модели может оказаться дорогостоящим и даже недопустимым по средствам, т.к. время. затраченное на планирование, повторное определение целей, выполнение анализа рисков и прототипирование, может быть чрезмерным;

6. Могут возникнуть затруднения при определении целей и стадий, указывающих на готовность продолжать процесс разработки на следующей итерации

Основная проблема спирального цикла – определение момента перехода на следующий этап. Для ее решения вводятся временные ограничения на каждый из этапов жизненного цикла, и переход осуществляется в соответствии с планом, даже если не вся запланированная работа закончена. Планирование производится на основе статистических данных, полученных в предыдущих проектах, и личного опыта разработчиков.

Вопрос 15. Основные процессы жизненного цикла и их спецификация.

Среди основных процессов жизненного цикла наибольшую важность имеют следующие: формирование требований и *ТЗ*, проектирование, разработка, тестирование, эксплуатация и сопровождение. Каждый процесс характеризуется определенными задачами и методами их решения, исходными данными, полученными на предыдущем этапе, и результатами.

Рассмотрим более подробно этапы:

1. Формирование ТЗ

Обычно этот этап включает в себя *предпроектное обследование*, где производится анализ предметной области, изучение объекта автоматизации,

сбор материалов, общение с заказчиками и формирование требований к разработке ИС.

2. Проектирование

Результатом проектирования является конкретное *проектное решение* с описанием компонент ИС, проработанная модель БД и т.д. На этом этапе происходит утверждение проектного решения для дальнейшей разработки ИС.

3. Разработка

Разработка информационной системы включает в себя все работы по созданию информационного программного обеспечения и его компонентов в соответствии с заданными требованиями. Процесс создания ИС не обходится без разработки сопроводительной документации, например:

• оформление проектной и эксплуатационной документации;

• подготовку материалов, необходимых для тестирования разработанных программных продуктов;

• разработку материалов, необходимых для обучения персонала.

4. Эксплуатация

Эксплуатационные работы можно подразделить на подготовительные и основные. К подготовительным относятся:

• конфигурирование базы данных и рабочих мест пользователей;

• обеспечение пользователей эксплуатационной документацией;

• обучение персонала.

Основные эксплуатационные работы включают:

• непосредственно эксплуатацию;

• локализацию проблем и устранение причин их возникновения;

• модификацию программного обеспечения;

• подготовку предложений по совершенствованию системы;

• развитие и модернизацию системы.

5. Сопровождение

Службы технической поддержки играют весьма заметную роль в жизни любой корпоративной информационной системы. Наличие квалифицированного технического обслуживания на этапе эксплуатации информационной системы является необходимым условием решения поставленных перед ней задач, причем ошибки обслуживающего персонала могут приводить к явным или скрытым финансовым потерям, сопоставимым со стоимостью самой информационной системы.

Основными предварительными действиями при подготовке к организации технического обслуживания информационной системы являются:

• выделение наиболее ответственных узлов системы и определение для них критичности простоя (это позволит выделить наиболее критичные составляющие информационной системы и оптимизировать распределение ресурсов для технического обслуживания);

• определение задач технического обслуживания и их разделение на внутренние, решаемые силами обслуживающего подразделения, и внешние, решаемые специализированными сервисными организациями (таким образом производится четкое определение круга исполняемых функций и разделение ответственности);

• проведение анализа имеющихся внутренних и внешних ресурсов, необходимых для организации технического обслуживания в рамках описанных задач и разделения компетенции (основные критерии для анализа: наличие гарантии на оборудование, состояние ремонтного фонда, квалификация персонала);

• подготовка плана организации технического обслуживания, в котором необходимо определить этапы исполняемых действий, сроки их исполнения, затраты на этапах, ответственность исполнителей.

Обеспечение качественного технического обслуживания информационной системы требует привлечения специалистов высокой квалификации, которые в состоянии решать не только каждодневные задачи администрирования, но и быстро восстанавливать работоспособность системы при сбоях.

Вопрос 16. Вспомогательные и организационные процессы жизненного цикла.

Среди вспомогательных процессов одно из главных мест занимает *управление конфигурацией (понятие определено в теме 1)*. Это тот вспомогательный процесс, который поддерживает основные процессы жизненного цикла информационной системы, прежде всего процессы разработки и сопровождения. При разработке проектов сложных информационных систем, состоящих из многих компонентов, каждый из которых может разрабатываться независимо и, следовательно, иметь несколько вариантов реализации и/или несколько версий одной реализации, возникает проблема учета их связей и функций, создания единой структуры и обеспечения развития всей системы. Управление конфигурацией позволяет организовывать, систематически учитывать и контролировать внесение изменений в различные компоненты информационной системы на всех стадиях ее жизненного цикла.

Управление проектом связано с вопросами планирования и организации работ, создания коллективов разработчиков и контроля за

сроками и качеством выполняемых работ. Именно поэтому в ЖЦ ИС существуют и организационные процессы, включающие в себя:

• выбор методов и инструментальных средств для реализации проекта;

• определение методов описания промежуточных состояний разработки;

• разработку методов и средств испытаний созданного программного обеспечения;

• обучение персонала.

Обеспечение качества проекта связано с проблемами верификации, проверки и тестирования компонентов информационной системы.

Верификация – это процесс определения соответствия текущего состояния разработки, достигнутого на данном этапе, требованиям этого этапа.

Проверка – это процесс определения соответствия параметров разработки исходным требованиям. Проверка отчасти совпадает с тестированием, которое проводится для определения различий между действительными и ожидавшимися результатами и оценки соответствия характеристик информационной системы исходным требованиям.

Вопрос 17. Стадии жизненного цикла.

В жизненном цикле выделяют следующие стадии:

1. *Предпроектное обследование (понятие определено в теме 4)*

Сбор материалов для проектирования:

- формирование требований;
- изучение объекта автоматизации;
- выбор и разработка варианта концепции системы.

Анализ материалов и разработка документации:

- создание и утверждение технико-экономического обоснования;
- разработка и, утверждение технического задания на проектирование информационной системы.

2. Проектирование

Предварительное проектирование:

- выбор проектных решений по всем аспектам разработки
- информационной системы;34
- описание всех компонентов информационной системы;
- оформление и утверждение технического проекта.

Детальное проектирование:

- выбор и разработка математических методов и алгоритмов
- программ;
- корректировка структур баз данных;

- создание документации на поставку и установку программных
- продуктов;
- выбор комплекса технических средств информационной,
- системы;
- создание документации на поставку и установку технических
- средств;
- разработка технорабочего проекта информационной системы.

3. Разработка информационной системы
   - получение и установка технических средств;
   - разработка, тестирование и доводка программ;
   - получение и установка программных средств;
   - разработка инструкций по эксплуатации программного обеспечения, технических средств, должностных инструкций для персонала.
4. Ввод информационной системы в эксплуатацию
   - ввод в опытную эксплуатацию технических средств;
   - ввод в опытную эксплуатацию программных средств;
   - обучение и сертифицирование персонала;
   - проведение опытной эксплуатации всех компонентов и системы в целом;
   - сдача в эксплуатацию и подписание актов приемки-сдачи работ.
5. Эксплуатация информационной системы
   - повседневная эксплуатация;
   - сопровождение программных, технических средств и всего проекта

Вопрос 18. управление качеством информационных систем и парадигма «философия качества информационных систем». Составляющие категории парадигмы.

При изучении определенной проблемы исходят из базового набора понятий, составляющего конструкцию содержания этой проблемы. В проблеме управления качеством информационных систем очень важно определить само понятие «управление качеством информационных систем». Поскольку пока нет нормированного определения указанного понятия, то здесь и далее мы будем пользоваться собственными определениями по данной проблеме. Вместе с тем в настоящее время созданы стандарты, которые содержат определения некоторых терминов, в той или иной мере связанных с проблемой управления качеством ИС и которые мы будет применять. В контексте рассматриваемых здесь понятий необходимо определить общее понимание качества ИС. Это общее представление может существовать в рамках парадигмы «философия качества информационных

систем». При определении предмета указанной парадигмы можно исходить из того, что в предмет, прежде всего, входят следующие категории:

1. Свойства, которые отображают сущность, содержание, значение ИС. Свойства представляются здесь как форма проявления сущности ИС – признаки, характеристики, параметры, взятые в их

взаимосвязи и взаимодействии.

2. Структура, как способ взаимосвязи и взаимодействия элементов. Элементы здесь отображаются набором видов, составляющих объем понятия – подпонятий, теорий, методов, средств теоретического и инструментального характера, предметов и процессов реального мира. Все эти элементы являются носителями свойств ИС.

3. Закономерности процессов, связанных с созданием и функционированием ИС и отображающих изменение ее качества. Процессы происходят во временном и пространственном измерениях. Формой временного отображения закономерностей принимается историческая шкала эволюции качества ИС. Формой пространственного проявления закономерностей может быть принята шкала взаимосвязи ИС с реальными объектами, отображающими как внешнюю, так и внутреннюю среды. Внешняя среда отношений ИС - это ее взаимодействие с надсистемой, а внутренняя - взаимодействие элементов составляющих структуру собственно самой ИС.

С учетом вышеприведенных категорий можно представить следующую трактовку понятия «философия качества информационных систем – это учение о качестве информационных систем, предметом которого является свойства, структура и закономерности процессов изменения качества информационных систем в разнообразии их отношений».

Вопрос 19. Понятие «качество информации» и понятие «качество ИС».

В данном случае целесообразно выделить понятие «качество информации», как опорное в парадигме «качество информационной системы». К этой парадигме можно отнести и понятие «качество данных», определяемое в производственно-техническом аспекте как «совокупность их свойств, обусловливающих их пригодность удовлетворять определенные потребности в соответствии с их назначением». Обратимся вновь к трем вышеуказанным категориям предмета «философии качества ИС». Если рассматривать информацию как объект Декартова трехмерного измерения, то ее можно представить следующими тремя категориями: «информация как сущность», «информация как свойство» и «информация как отношение». Каждая из указанных категорий занимает одну из плоскостей Декартова

пространства. Они находятся в относительно четкой координации и субординации друг к другу. Это значит, что в определенных условиях существования информации эти категории и их типологические элементы могут взаимопроникать и взаимозаменять друг друга. Так, например, сущность информации отображается принадлежащими ей свойствами. Эти свойства могут создавать иерархическую структуру, то есть находиться в отношении субординации и координации. С позиций семиотики разнообразие свойств информационных объектов может быть сформировано в семантические, синтаксические и прагматические группы свойств. Они отображают проявление связи и взаимодействие информации в плоскости ее отношений как на внутреннем, так и на внешнем уровнях – элементном, технологическом, производственном, экономическом, товарном, потребительском, социальном, аксеологическом и т.д. Отсюда «качество информации – это совокупность свойств информации, отражающая сущность информации в разнообразии её отношений». Система отношений понятия «качество информации» обусловливает существование «качества информационной системы» как особой формы проявления своих отношений с реальным миром. Отсюда конструкция определения «качество информационной системы» будет мало отличаться от конструкции «качество информации». В методологическом плане можно констатировать, что качество информационной системы – это совокупность свойств, составляющая сущность информационной системы в разнообразии ее отношений. Таким образом, «качество информационной системы – это совокупность свойств, составляющая сущность информационной системы и ее способность удовлетворять установленные требования в соответствии со своим назначением»

Вопрос 20. Процедура оценки качества ИС.

Процедуры оценки качества ИС базируются на применении средств научной дисциплины - *квалиметрии* (от лат. «qualis- качество, metreo – измерение). В этом плане квалиметрия может рассматриваться как часть эконометрики и одновременно метрологии. Поскольку категория «качество» является универсальной, то квалиметрия – это междисциплинарная наука, относящаяся к технике, к экономике, образованию, здравоохранению, информатике и др. В квалиметрии информационных систем значительное место занимает задача измерения качества ИС. В основе адекватной оценки лежит набор процедур по корректному измерению качества ИС. В этом измерении применяется арсенал методов и средств. Измерению подвергаются свойства ИС, вернее интенсивность их проявления. При этом измерение производится в соответствующих единицах.

Таким образом, «*измерение качества информационной системы* - это совокупность процедур, выполняемых при помощи средств измерений с целью нахождения числовых значений свойств информационной системы в принятых единицах измерения». В принципиальном отношении методы оценки качества различных объектов, используемые в прикладных разделах квалиметрии, практически одинаковы. В тривиальной форме оценка качества ИС может быть выполнена по трем основным уровням: выше требований, соответствует требованиям, ниже требований. В практике управления качеством ИС оценка качества выполняется на основе квантификации свойств ИС. Каждое свойство может быть представлено в форме показателя качества, значение которого представляется в цифровой форме. На основе сопоставительного анализа определяется уровень соответствия ИС определенным требованиям, которым она должна удовлетворять. В контексте рассмотренных выше понятий можно принять, что «*оценка качества информационной системы* – это реализация комплекса методов и средств по определению уровня соответствия информационной системы установленным требованиям».

Вопрос 21. Понятие проекта и особенности проектной работы.

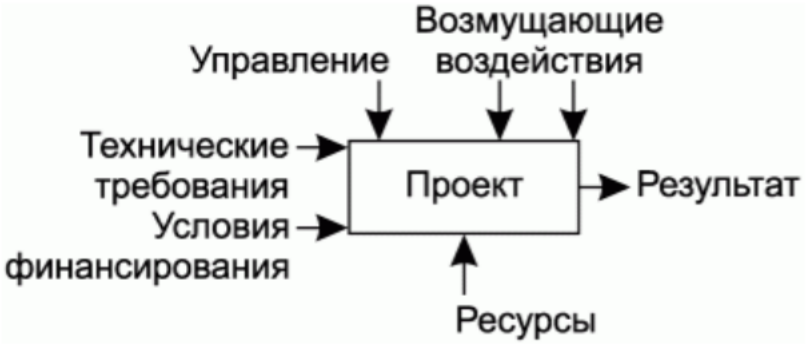
Проект – попытка действий с определенными начальными и конечными сроками, предпринимаемая для создания продукта или услуги в соответствии с заданными ресурсами и требованиями

Можно выделить следующие основные отличительные признаки проекта как объекта управления:

- изменчивость – целенаправленный перевод системы из существующего в некоторое желаемое состояние, описываемое в терминах целей проекта;
- ограниченность конечной цели;
- ограниченность продолжительности;
- ограниченность бюджета;
- ограниченность требуемых ресурсов;
- новизна для предприятия, для которого реализуется проект;
- комплексность – наличие большого числа факторов, прямо или косвенно влияющих на прогресс и результаты проекта;
- правовое и организационное обеспечение – создание специфической организационной структуры на время реализации проекта.

Вопрос 22. Процесс управления проектами.

В системном плане проект может быть представлен «черным ящиком», на входе которого располагаются технические требования и условия финансирования, а на выходе – требуемый результат.



Выполнение работ обеспечивается наличием необходимых ресурсов:

- материалов;
- оборудования;
- человеческих ресурсов.

Эффективность работ достигается за счет управления процессом выполнения проекта, что обеспечивает распределение ресурсов, координацию выполняемой последовательности работ и компенсацию внутренних и внешних возмущающих воздействий.

С точки зрения теории систем управления проект как объект управления должен быть наблюдаемым и управляемым, то есть выделяются некоторые характеристики, по которым можно постоянно контролировать ход выполнения проекта (свойство наблюдаемости). Кроме того, необходимы механизмы своевременного воздействия на ход выполнения проекта (свойство управляемости). Свойство управляемости особенно актуально в условиях неопределенности и изменчивости предметной области, которые нередко сопутствуют проектам по разработке информационных систем (более подробно проблемы получения полного формального описания предметной области будут обсуждаться в конце данной главы).

Для обоснования целесообразности и осуществимости проекта, анализа хода его реализации, а также для заключительной оценки степени достижения поставленных целей проекта и сравнения фактических результатов с запланированными существует ряд характеристик проекта.

К важнейшим из них относятся технико-экономические показатели:

- объем работ;
- сроки выполнения;
- себестоимость;
- экономическая эффективность, обеспечиваемая реализацией проекта;
- социальная и общественная значимость проекта.

Вопрос 23. Классификация проектов.

Проекты могут значительно отличаться по сфере приложения, составу, предметной области, масштабам, длительности, составу участников, степени сложности, значимости результатов и т. п.

Проекты могут быть классифицированы по самым различным признакам. Отметим основные из них.

• *Класс проекта* определяется по составу и структуре проекта.

- ❖ монопроект (отдельный проект, который может быть любого типа, вида и масштаба);
- ❖ мультипроект (комплексный проект, состоящий из ряда монопроектов и требующий мультипроектного управления).

• *Тип проекта* определяется по основным сферам деятельности, в которых осуществляется проект. Можно выделить пять основных типов проекта:

- ❖ технический;
- ❖ организационный;
- ❖ экономический;
- ❖ социальный;
- ❖ смешанный.

Примечание. Разработка информационных систем относится, скорее всего, к техническим проектам. У таких проектов две особенности. Во-первых,

главная цель проекта четко определена, но отдельные цели должны уточняться по мере достижения частных результатов. Во-вторых, срок завершения и продолжительность проекта определены заранее, желательно их точное соблюдение, однако они также могут корректироваться в зависимости от полученных промежуточных результатов и общего прогресса проекта.

• *Масштаб проекта* определяется размером бюджета и количеством участников: –

- ❖ мелкие проекты;
- ❖ малые проекты;
- ❖ средние проекты;
- ❖ крупные проекты.

Можно также рассматривать масштабы проектов в более конкретной форме – отраслевые, корпоративные, ведомственные проекты, проекты одного предприятия.

Вопрос 24. Основные фазы проектирования информационной системы и их особенности.

Каждый проект, независимо от сложности и объема работ, необходимых для его выполнения, проходит в своем развитии определенные состояния: от состояния, когда «проекта еще нет», до состояния, когда «проекта уже нет». Совокупность ступеней развития от возникновения идеи до полного завершения проекта принято разделять на фазы (стадии, этапы).

В определении количества фаз и их содержания имеются некоторые отличия, поскольку эти характеристики во многом зависят от условий осуществления конкретного проекта и опыта основных участников. Тем не менее, логика и основное содержание процесса разработки информационной системы почти во всех случаях являются общими.

Можно выделить следующие фазы развития информационной системы:

1. формирование концепции;
2. подготовка технического задания;
3. проектирование;
4. разработка;

5. ввод системы
6. эксплуатацию.

Рассмотрим каждую из них более подробно. Примечание. Вторую и частично третью фазы принято называть фазами системного проектирования, а последние две (иногда сюда включают и фазу проектирования) – фазами реализации.

1. *Формирование концепции*.

Главным содержанием работ на концептуальной фазе является определение проекта, разработка его концепции, включающая:

- формирование идеи, постановку целей;
- формирование ключевой команды проекта;
- изучение мотивации и требований заказчика и других участников;
- сбор исходных данных и анализ существующего состояния;
- определение основных требований и ограничений, требуемых материальных, финансовых и трудовых ресурсов;
- сравнительную оценку альтернатив;
- представление предложений, их экспертизу и утверждение.

2. *Подготовка технического задания*

Главным содержанием фазы подготовки технического задания является уточнение технического предложения в ходе переговоров с заказчиком о заключении контракта.

Общее содержание работ этой фазы:

- разработка основного содержания, базовой структуры проекта;
- разработка и утверждение технического задания;
- планирование, декомпозиция базовой структурной модели проекта;
- составление сметы и бюджета проекта, определение потребности в ресурсах;
- разработка календарных планов и укрупненных графиков работ;
- подписание контракта с заказчиком;
- ввод в действие средств коммуникации участников проекта и средств контроля за ходом работ.

3. *Проектирование*

На фазе проектирования определяются подсистемы, их взаимосвязи, выбираются наиболее эффективные способы выполнения проекта и использования ресурсов.

Характерные работы этой фазы:

- выполнение базовых проектных работ;

- разработка частных технических заданий;
- выполнение концептуального проектирования;
- составление технических спецификаций и инструкций;
- представление проектной разработки, экспертиза и утверждение.

4. *Разработка*

На фазе разработки производятся координация и оперативный контроль работ по проекту, осуществляется изготовление подсистем, их объединение и тестирование.

Основное содержание:

- выполнение работ по разработке программного обеспечения;
- подготовка к внедрению системы;
- контроль и регулирование основных показателей проекта.

5. *Ввод системы в эксплуатацию*

На фазе ввода системы в эксплуатацию проводятся испытания, идет опытная эксплуатация системы в реальных условиях, ведутся переговоры о результатах выполнения проекта и о возможных новых контрактах.

Основные виды работ:

- комплексные испытания;
- подготовка кадров для эксплуатации создаваемой системы;
- подготовка рабочей документации, сдача системы заказчику и ввод ее в эксплуатацию; сопровождение, поддержка, сервисное обслуживание;
- оценка результатов проекта и подготовка итоговых документов;
- разрешение конфликтных ситуаций и закрытие работ по проекту;
- накопление опытных данных для последующих проектов, анализ опыта, состояния, определение направлений развития.

Начальные фазы проекта имеют решающее влияние на достигаемый результат, так как в них принимаются основные решения, определяющие качество информационной системы. При этом обычно 30 % вклада в конечный результат проекта вносят фазы концепции и предложения, 20 % – фаза проектирования, 20 % – фаза разработки, 30 % – фаза сдачи объекта и завершения проекта.

Следует иметь в виду, что на обнаружение ошибок, допущенных на стадии системного проектирования, расходуется примерно в два раза больше времени, чем на последующих фазах, а их исправление обходится в пять раз дороже. Поэтому на начальных стадиях проекта разработку следует выполнять особенно тщательно.

Наиболее часто на начальных фазах допускаются следующие ошибки:

- ошибки в определении интересов заказчика;
- концентрация на маловажных, сторонних интересах;
- неправильная интерпретация исходной задачи;
- неправильное или недостаточное понимание деталей;
- неполнота функциональных спецификаций (системных требований);
- ошибки в определении требуемых ресурсов и сроков;
- редкая проверка на согласованность этапов и отсутствие контроля со стороны заказчика (нет привлечения заказчика).

Вопрос 25. Задачи методологии проектирования ИС.

Основными задачами, решение которых должна обеспечивать методология создания корпоративных информационных систем (с помощью соответствующего набора инструментальных средств), являются:

- соответствие создаваемой информационной системы целям и задачам предприятия, а также предъявляемым к ней требованиям по автоматизации бизнес-процессов;
- гарантирование создания системы с заданными параметрами в течение заданного времени в рамках оговоренного заранее бюджета;
- простота сопровождения, модификации и расширения системы с целью обеспечения ее соответствия изменяющимся условиям работы предприятия;
- соответствие создаваемой корпоративной информационной системы требованиям открытости, переносимости и масштабируемости;
- возможность использования в создаваемой системе разработанных ранее и применяемых на предприятии средств информационных технологий (программного обеспечения, баз данных, средств вычислительной техники, телекоммуникаций).

Вопрос 26. Технология проектирования ИС: составляющие части, ресурсы, требования к технологии.

Основное содержание технологии проектирования составляют технологические инструкции, состоящие из описания последовательности технологических операций, условий, в зависимости от которых выполняется та или иная операция, и описаний самих операций.

Технология проектирования может быть представлена как **совокупность трех составляющих**:

- заданной последовательности выполнения технологических операций проектирования;
- критериев и правил, используемых для оценки результатов выполнения технологических операций;
- графических и текстовых средств (нотаций), используемых для описания проектируемой системы.

Каждая технологическая операция должна обеспечиваться следующими материальными и информационными **ресурсами**:

- данными, полученными на предыдущей операции (или исходными данными), представленными в стандартном виде;
- методическими материалами, инструкциями, нормативами и стандартами;
- программными и техническими средствами;
- исполнителями.

Результаты выполнения операции должны представляться в некотором стандартном виде, обеспечивающем их адекватное восприятие при выполнении следующей технологической операции (на которой они будут использоваться в качестве исходных данных).

Можно сформулировать **ряд общих требований**, которым должна удовлетворять технология проектирования, разработки и сопровождения информационных систем:

- поддерживать полный жизненный цикл информационной системы;
- обеспечивать гарантированное достижение целей разработки системы с заданным качеством и в установленное время;
- обеспечивать возможность разделения (декомпозиции) крупных проектов на ряд подсистем – составных частей, разрабатываемых группами исполнителей ограниченной численности, с последующей интеграцией этих частей;

  Примечание.

  Декомпозиция проекта позволяет повысить эффективность работ. Подсистемы, на которые разбивается проект, должны быть слабо связаны по данным и функциям. Каждая подсистема

разрабатывается отдельной группой разработчиков. При этом необходимо обеспечить координацию работ и исключить дублирование результатов, получаемых каждой проектной группой.

- обеспечивать возможность ведения работ по проектированию отдельных подсистем небольшими группами (3–7 человек), что обусловлено принципами управляемости коллектива и повышения производительности за счет минимизации числа внешних связей;
- обеспечивать минимальное время получения работоспособной системы;

  Примечание.

  Здесь имеется в виду не реализация информационной системы в целом, а разработка ее отдельных подсистем. Как правило, даже при наличии полностью завершенного проекта внедрение разработанной системы проводится последовательно отдельными подсистемами. Реализация же всей системы в сжатые сроки может потребовать привлечения большого числа разработчиков, при этом эффект может оказаться менее значимым, чем при реализации отдельных подсистем в более короткие сроки меньшим числом разработчиков.
- предусматривать возможность управления конфигурацией проекта, ведения версий проекта и его составляющих, автоматического выпуска проектной документации и синхронизацию ее версий с версиями проекта;
- обеспечивать независимость выполняемых проектных решений от средств реализации системы – системы управления базами данных, операционной системы, языка и системы программирования.

Вопрос 27. Методология *RAD* и ее основные особенности.

На начальном этапе существования компьютерных информационных систем их разработка велась на традиционных языках программирования. Однако по мере возрастания сложности разрабатываемых систем и запросов пользователей (чему в значительной степени способствовал прогресс в области вычислительной техники, а также появление удобного графического интерфейса пользователя в системном программном обеспечении)

потребовались новые средства, обеспечивающие значительное сокращение сроков разработки.

Это послужило предпосылкой к созданию целого направления в области программного обеспечения – инструментальных средств для быстрой разработки приложений. Развитие этого направления привело к появлению на рынке программного обеспечения средств автоматизации практически всех этапов жизненного цикла информационных систем.

Основные особенности методологии RAD

Методология создания информационных систем, основанная на использовании средств быстрой разработки приложений, получила в последнее время широкое распространение и приобрела название методологии быстрой разработки приложений (Rapid Application Development, RAD).

Данная методология охватывает все этапы жизненного цикла современных информационных систем. Методология RAD – это комплекс специальных инструментальных средств, позволяющих оперировать с определенным набором графических объектов, функционально отображающих отдельные информационные компоненты приложений.

Вопрос 28. RAD как инструмент разработки приложений.

Под методологией быстрой разработки приложений обычно понимается процесс разработки информационных систем, основанный на трех основных элементах:

- небольшой команде программистов (обычно от 2 до 10 человек);
- тщательно проработанном производственном графике работ, рассчитанном на сравнительно короткий срок разработки (от 2 до 6 мес.);
- итерационной модели разработки, основанной на тесном взаимодействии с заказчиком – по мере выполнения проекта разработчики уточняют и реализуют в продукте требования, выдвигаемые заказчиком.

При использовании методологии RAD большое значение имеют опыт и профессионализм разработчиков. Группа разработчиков должна состоять из профессионалов, имеющих опыт в анализе, проектировании, программировании и тестировании программного обеспечения.

Основные принципы методологии RAD можно свести к следующим:

- используется итерационная (спиральная) модель разработки;
- полное завершение работ на каждом из этапов жизненного цикла не обязательно;
- в процессе разработки информационной системы обеспечивается тесное взаимодействие с заказчиком и будущими пользователями;
- применяются CASE-средства и средства быстрой разработки приложений;
- применяются средства управления конфигурацией, облегчающие внесение изменений в проект и сопровождение готовой системы;
- используются прототипы, позволяющие полнее выяснить и реализовать потребности конечного пользователя;
- тестирование и развитие проекта осуществляются одновременно с разработкой;
- разработка ведется немногочисленной и хорошо управляемой командой профессионалов;
- обеспечиваются грамотное руководство разработкой системы, четкое планирование и контроль выполнения работ.

Вопрос 29. Объектно-ориентированный подход к созданию приложений.

Средства RAD позволили реализовать совершенно иную по сравнению с традиционной технологию создания приложений: информационные объекты формируются как некие действующие модели (прототипы), чье функционирование согласуется с пользователем, а затем разработчик может переходить непосредственно к формированию законченных приложений, не теряя из виду общей картины проектируемой системы.

Возможность использования подобного подхода в значительной степени является результатом применения принципов объектно-ориентированного проектирования. Эти принципы позволяют преодолеть одну из главных

трудностей, возникающих при разработке сложных систем, – колоссальный разрыв между реальным миром (предметной областью) и имитирующей средой.

Объектно-ориентированное программирование (ООП, Object-Oriented Programming) - совокупность принципов, технологий, а также инструментальных средств для создания программных систем на основе архитектуры взаимодействия *объектов*.

Особенности

Использование объектно-ориентированных принципов позволяет создать описание (модель) предметной области в виде совокупности объектов – сущностей, объединяющих данные и методы обработки этих данных (процедуры). Каждый объект обладает собственным поведением и моделирует некоторый объект реального мира. С этой точки зрения объект является вполне осязаемым и демонстрирует определенное поведение.

В объектном подходе акцент переносится на конкретные характеристики физической или абстрактной системы, являющейся предметом программного моделирования. Объекты обладают целостностью, которая не может быть нарушена. Таким образом, свойства, характеризующие объект и его поведение, остаются неизменными. Объект может только менять состояние, управляться или становиться в определенное отношение к другим объектам.

Где используется

Широкое распространение объектно-ориентированное программирование получило с появлением средств визуального программирования, которые обеспечивают слияние (инкапсуляцию) данных с процедурами, описывающими поведение реальных объектов, в объекты программ, которые могут быть отображены определенным образом в графической пользовательской среде. Это позволило приступить к созданию программных систем, максимально похожих на реальные, и добиваться наивысшего уровня абстракции. В свою очередь, объектно-ориентированное программирование позволяет создавать более надежные коды, так как у объектов программ существует точно определенный и жестко контролируемый интерфейс.

Объектно-ориентированный подход в RAD

При разработке приложений с помощью инструментов RAD используются множество готовых объектов, сохраняемых в общедоступном хранилище. Однако обеспечивается и возможность разработки новых объектов. При этом новые объекты могут разрабатываться как на основе существующих, так и «с нуля».

Инструментальные средства RAD обладают удобным графическим интерфейсом пользователя и позволяют на основе стандартных объектов формировать простые приложения без написания кода программы. Это является большим преимуществом RAD, так как в значительной степени сокращает рутинную работу по разработке интерфейсов пользователя (при использовании обычных средств разработка интерфейсов представляет собой достаточно трудоемкую задачу, отнимающую много времени).

Высокая скорость разработки интерфейсной части приложений позволяет быстро создавать прототипы и упрощает взаимодействие с конечными пользователями. Таким образом, инструменты RAD позволяют разработчикам сконцентрировать усилия на сущности реальных бизнес-процессов предприятия, для которого создается информационная система. В итоге это приводит к повышению качества разрабатываемой системы.

Вопрос 30. Визуальное программирование.

Применение принципов объектно-ориентированного программирования позволило создать принципиально новые средства проектирования приложений, называемые средствами визуального программирования.

Особенности

Визуальные инструменты RAD позволяют создавать сложные графические интерфейсы пользователя вообще без написания кода программы. При этом разработчик может на любом этапе наблюдать то, что закладывается в основу принимаемых решений.

Визуальные средства разработки оперируют в первую очередь со стандартными интерфейсными объектами – окнами, списками, текстами, которые легко можно связать с данными из базы данных и отобразить на экране монитора. Другая группа объектов представляет собой стандартные

элементы управления – кнопки, переключатели, флажки, меню и т. п., с помощью которых осуществляется управление отображаемыми данными. Все эти объекты могут быть стандартным образом описаны средствами языка, а сами описания сохранены для многократного использования в будущем.

В настоящее время существует довольно много различных визуальных *средств разработки* приложений, но все они могут быть разделены на две группы:

- Среди универсальных систем визуального программирования сейчас наиболее распространены такие, как Borland Delphi и Visual Basic.

Универсальными мы их называем потому, что они не ориентированы исключительно на разработку приложений баз данных – с их помощью могут быть разработаны приложения почти любого типа, в том числе и информационные приложения. Причем программы, разрабатываемые с помощью универсальных систем, могут взаимодействовать практически с любыми системами управления базами данных.

- Специализированные средства разработки ориентированы только на создание приложений баз данных. Причем, как правило, они привязаны к вполне определенным системам управления базами данных. В качестве примера таких систем можно привести Power Builder фирмы Sybase (естественно, предназначенный для работы с СУБД Sybase Anywhere Server) и Visual FoxPro фирмы Microsoft.

Поскольку задачи создания прототипов и разработки пользовательского интерфейса по существу слились, программист получил непрерывную обратную связь с конечными пользователями, которые могут не только наблюдать за созданием приложения, но и активно участвовать в нем, корректировать результаты и свои требования. Это также способствует сокращению сроков разработки и является важным психологическим аспектом, который привлекает к RAD все большее число разработчиков.

Визуальные инструменты RAD позволяют максимально сблизить этапы создания информационных систем: анализ исходных условий, проектирование системы, разработка прототипов и окончательное формирование приложений становятся сходными, так как на каждом этапе разработчики оперируют визуальными объектами.

Вопрос 31. Событийное программирование.

Логика приложения, построенного средствами RAD, является событийно-ориентированной. Это означает, что каждый объект, входящий в состав приложения, может генерировать события и реагировать на события, генерируемые другими объектами. Примерами событий могут быть открытие и закрытие окон, щелчок на кнопке, нажатие клавиши клавиатуры, движение мыши, изменение данных в базе данных и т. п.

Особенности

Разработчик реализует логику приложения путем определения обработчика каждого события – процедуры, выполняемой объектом при наступлении соответствующего события. Например, обработчик события «щелчок на кнопке» может открыть диалоговое окно. Таким образом, управление объектами осуществляется с помощью событий.

Обработчики событий, связанных с управлением базой данных (DELETE, INSERT, UPDATE), могут реализовываться в виде триггеров на клиентском или серверном узле. Такие обработчики позволяют обеспечить ссылочную целостность базы данных при выполнении операций удаления, вставки и обновления, а также автоматическую генерацию первичных ключей.

Поэтому можно утверждать, что сложные проекты, разрабатываемые по каскадной схеме, имеют повышенный уровень риска. Этот вывод подтверждается практикой: по сведениям консалтинговой компании The Standish Group в США более 31 % проектов корпоративных информационных систем (IT-проектов) заканчивается неудачей; почти 53 % IT-проектов завершается с перерасходом бюджета (в среднем на 189 %, то есть почти в два раза); и только 16,2 % проектов укладывается и в срок, и в бюджет.

Помимо рассмотренных существует еще один серьезный недостаток, присущий каскадной модели разработки, на который также следует обратить внимание. Этот недостаток связан с конфликтом (не всегда явным) между разработчиками, участвующими в выполнении проекта. Конфликт обусловлен тем, что возврат части проекта на предыдущую 60 стадию

обычно сопровождается поиском причин и виновных. А так как однозначно персонифицировать ответственного за ошибки можно далеко не всегда, попытки поиска виноватых могут сильно усложнить отношения в коллективе.

Как следствие, в рабочей группе часто ценится не тот руководитель, который имеет высокую квалификацию и больший опыт, а тот, кто умеет «отстоять» своих подчиненных, обеспечить им более удобные условия работы и т. п.

В результате появляется опасность снижения и квалификации, и творческого потенциала всей команды. Соответственно, техническое руководство проектом начинает в большей степени подменяться организационным руководством, все более детальной проработкой должностных инструкций и все более формальным их исполнением. Тот, кто не умеет организовать работу, обречен бороться за дисциплину. И здесь возникает проблема несовместимости дисциплины и творчества. Чем строже дисциплина, тем менее творческой становится атмосфера в коллективе. Такое положение вещей может привести к тому, что наиболее одаренные кадры со временем покинут коллектив.

Вопрос 32. Фазы жизненного цикла в рамках методологии *RAD.*

При использовании методологии быстрой разработки приложений жизненный цикл информационной системы состоит из четырех фаз:

1. анализа и планирования требований;

2. проектирования;

3. построения;

4. внедрения.

1. Фаза анализа и планирования требований

На фазе анализа и планирования требований *определяются:*

- функции, которые должна выполнять разрабатываемая информационная система;

- наиболее приоритетные функции, требующие разработки в первую очередь;
- информационные потребности (Определение указанных выше требований выполняется совместно будущими пользователями системы и разработчиками.)
- масштаб проекта;
- временные рамки для каждой из последующих фаз;
- сама возможность реализации данного проекта в установленных рамках финансирования на имеющихся аппаратных и программных средствах.

Если реализация проекта принципиально возможна, то *результатом фазы* анализа и планирования требований будет список функций разрабатываемой информационной системы с указанием их приоритетов, а также предварительные функциональные и информационные модели системы.

2. Фаза проектирования

1. Создание прототипов приложений с использованием CASE-средств. Прототипы, созданные с помощью CASE-средств, анализируются пользователями, которые уточняют и дополняют те требования к системе, которые не были выявлены на предыдущей фазе. Таким образом, *на данной фазе также необходимо участие будущих пользователей в техническом проектировании системы*.

2. Далее на этой фазе *проводится анали*з и, если требуется, *корректировка* функциональной модели системы. Детально рассматривается каждый процесс системы. При необходимости для каждого элементарного процесса создается *частичный прототип*: экран, диалоговое окно или отчет (это позволяет устранить неясности или неоднозначности).

3. Затем *определяются требования разграничения доступа к данным*.

Примечание. Для построения всех моделей и прототипов должны быть использованы именно те CASE-средства, которые будут затем применяться при построении системы. Данное требование связано с тем, что при передаче информации о проекте с этапа на этап может произойти фактически неконтролируемое искажение данных.

Применение единой среды хранения проектной информации позволяет избежать этой опасности.

4. После детального рассмотрения процессов *определяется количество функциональных элементов разрабатываемой системы*. Это позволяет разделить информационную систему на ряд подсистем, каждая из которых реализуется одной командой разработчиков за приемлемое для RAD-проектов время (порядка полутора месяцев). С использованием CASE-средств проект распределяется между различными командами – делится функциональная модель. На этой же фазе происходит определение набора необходимой документации.

Результатами данной фазы являются:

- общая информационная модель системы;
- функциональные модели системы в целом и подсистем, реализуемых отдельными командами разработчиков;
- точно определенные с помощью CASE-средства интерфейсы между автономно разрабатываемыми подсистемами;
- построенные прототипы экранов, диалоговых окно и отчетов.

Примечание.

Одной из особенностей применения методологии RAD на данной фазе является то, что каждый созданный прототип развивается в часть будущей системы. Таким образом, на следующую фазу передается более полная и полезная информация. В противоположность этому при традиционном подходе использовались средства прототипирования, не предназначенные для построения реальных приложений, поэтому разработанные прототипы не годились для последующих фаз и просто «выбрасывались» после того, как решали задачу устранения неясностей в проекте.

3. Фаза построения. На фазе построения выполняется собственно быстрая разработка приложения.

Разработчики производят *итеративное построение реальной системы на основе полученных ранее моделей*, а также *требований нефункционального характера*. Разработка приложения ведется средствами визуального программирования. Формирование программного кода частично

выполняется с помощью автоматических генераторов кода, входящих в состав CASE-средств. Код генерируется на основе разработанных моделей.

На фазе построения также *требуется участие пользователей системы*, которые оценивают получаемые результаты и *вносят коррективы*, если в процессе разработки система перестает удовлетворять определенным ранее требованиям.

Тестирование системы осуществляется непосредственно в процессе разработки.

После окончания работ каждой отдельной команды разработчиков производится постепенная *интеграция данной части системы с остальными*, формируется полный программный код, выполняется тестирование совместной работы данной части приложения с остальными, а затем тестирование системы в целом.

Завершается физическое проектирование системы, а именно:

- определяется необходимость распределения данных;
- производится анализ использования данных;
- производится физическое проектирование базы данных;
- определяются требования к аппаратным ресурсам;
- определяются способы повышения производительности;
- завершается разработка документации проекта.

Результатом данной фазы является готовая информационная система, удовлетворяющая всем требованиям пользователей.

4. Фаза внедрения

Фаза внедрения в основном *сводится к обучению пользователей разработанной информационной системы*. Так как фаза построения достаточно непродолжительна, планирование и подготовка к внедрению должны начинаться заранее, еще на этапе проектирования системы.

Примечание.

Приведенная схема разработки информационной системы не является универсальной. Вполне возможны различные отклонения от нее. Это связано

с зависимостью схемы выполнения проекта от начальных условий, при которых начинается разработка (например, разрабатывается совершенно новая система или на предприятии уже существует некоторая информационная система). Во втором случае существующая система может либо использоваться в качестве прототипа новой системы, либо интегрироваться в новую разработку в качестве одной из подсистем.

Вопрос 33. Недостатки и ограничения методологии RAD.

Несмотря на все свои достоинства, методология RAD (как, впрочем, и любая другая методология), не может претендовать на универсальность.

Недостатки

Ее применение *наиболее эффективно при создании сравнительно небольших систем, разрабатываемых для конкретного заказчика*. При разработке же типовых систем, не являющихся законченным продуктом, а представляющих собой совокупность типовых элементов информационной системы, большое значение имеют такие показатели проекта, как управляемость и качество, которые могут войти в противоречие с простотой и скоростью разработки. Это связано с тем, что типовые системы обычно централизованно сопровождаются и могут адаптироваться к различным программно-аппаратным платформам, системам управления базами данных, коммуникационным средствам, а также интегрироваться с существующими разработками. Поэтому для такого рода проектов необходим высокий уровень планирования и жесткая дисциплина проектирования, строгое следование заранее разработанным протоколам и интерфейсам, что снижает скорость разработки.

Ограничения

- Методология RAD не подходит для создания не только типовых информационных систем, но и сложных расчетных программ, операционных систем и программ управления сложными инженерно-техническими объектами, то есть программ, требующих написания *большого объема уникального кода*.
- Методология RAD не может быть использована для разработки приложений, в которых интерфейс пользователя является вторичным, то есть *отсутствует наглядное определение логики работы системы*. Примерами таких приложений могут служить приложения реального времени, драйверы или службы.

- Совершенно неприемлема методология RAD для разработки *систем, от которых зависит безопасность людей,* например, систем управления транспортом или атомными электростанциями. Это обусловлено тем, что итеративный подход, являющийся одной из основ RAD, предполагает, что первые версии системы не будут полностью работоспособными, что в данном случае может привести к серьезнейшим катастрофам.

Вопрос 34. Понятие и применение открытых ИС.

Создание, сопровождение и развитие современных сложных информационных систем базируется на методологии построения таких систем как открытых. Открытая информационная система - система, которая реализует открытые спецификации на интерфейсы, сервисы (услуги среды) и поддерживаемые форматы данных, достаточные для того, чтобы дать возможность должным образом разработанному прикладному программному обеспечению быть переносимым в широком диапазоне систем с минимальными изменениями, взаимодействовать с другими приложениями на локальных и удалённых системах, и взаимодействовать с пользователями в стиле, который облегчает переход пользователей от системы к системе.

Открытые информационные системы создаются в процессе информатизации всех основных сфер современного общества: органов государственного управления, финансово-кредитной сферы, информационного обслуживания предпринимательской деятельности, производственной сферы, науки, образования. Развитие и использование открытых информационных систем неразрывно связаны с применением стандартов на основе методологии функциональной стандартизации информационных технологий.

Вопрос 35. Понятие профиля ИС.

При создании и развитии сложных, распределенных, тиражируемых информационных систем требуется гибкое формирование и применение гармонизированных совокупностей базовых стандартов и нормативных документов разного уровня, выделение в них требований и рекомендаций, необходимых для реализации заданных функций системы. Для унификации и регламентирования такие совокупности базовых стандартов должны адаптироваться и конкретизироваться применительно к определенным классам проектов, функций, процессов и компонентов системы. В связи с этим выделилось и сформировалось понятие профиля информационной системы как основного инструмента функциональной стандартизации.

Профиль ИС – это совокупность нескольких (или подмножество одного) базовых стандартов с четко определенными и гармонизированными

подмножествами обязательных и факультативных возможностей, предназначенная для реализации заданной функции или группы функций.

Вопрос 36. Классификация профилей ИС.

Обычно рассматривают две группы профилей, регламентирующих:

1. Архитектуру и структуру информационной системы;

2. Процессы проектирования, разработки, применения, сопровождения и развития системы.

В зависимости от области применения профили могут иметь разные категории и соответственно разные статусы утверждения:

1. Профили конкретной информационной системы, определяющие стандартизованные проектные решения в пределах данного проекта;

2. Профили информационной системы, предназначенные для решения некоторого класса прикладных задач.

Вопрос 37. Принципы формирования профиля ИС.

Профили информационных систем призваны решить следующие задачи:

1. Снижение трудоемкости проектов;

2. Повышение качества компонентов информационных систем;

3. Обеспечение расширяемости и масштабируемости разрабатываемых систем;

4. Обеспечение возможности функциональной интеграции в информационную систему задач, которые раньше решались раздельно;

5. Обеспечение переносимости прикладного программного обеспечения.

В зависимости от того, какие из указанных задач являются наиболее приоритетными, производится выбор стандартов и документов для формирования профиля.

В качестве общей методологической базы построения и применения профилей сложных распределенных ИС предлагается использовать технический отчет ИСО/МЭК ТО 10000 (ISO/IEC TR 10000-1, ISO/IEC TR 10000-2, ISO/IEC TR 10000-3). Части 1 и 2 этого документа введены в России в качестве стандарта ГОСТ Р. Часть 3, определяющую основы и таксономию профилей среды открытых систем, предлагается задействовать при построении и использовании профилей ИС как документ прямого применения.

Вопрос 38. Актуальность использования профилей ИС.

Актуальность использования профилей информационных систем обусловлена современным состоянием стандартизации информационных технологий, которое характеризуется следующими особенностями:

1. Существует множество международных и национальных стандартов, которые не полностью и неравномерно удовлетворяют потребности в стандартизации объектов и процессов создания и применения сложных информационных систем;

2. Длительные сроки разработки, согласования и утверждения международных и национальных стандартов приводят к их консерватизму и хроническому отставанию от современных информационных технологий;

3. Функциональными стандартами поддержаны и регламентированы только самые простые объекты и рутинные, массовые процессы (телекоммуникации, программирование, документирование программ и данных), а наиболее сложные и творческие процессы создания и развития крупных распределенных информационных систем (системный анализ и проектирование, интеграция компонентов и систем, испытания и сертификация) почти не поддержаны требованиями и рекомендациями стандартов из-за трудности их формализации и унификации;

4. Совершенствование и согласование нормативных и методических документов в ряде случаев позволяют создать на их основе национальные и международные стандарты.

Вопрос 39. Подходы к формированию профилей ИС.

Подходы к формированию профилей информационных систем могут быть различными. В международной функциональной стандартизации информационных технологий принято довольно жесткое понятие профиля. Считается, что его основой могут быть только утвержденные международные и национальные стандарты. Использование стандартов де-факто и нормативных фирменных документов не допускается. При таком подходе затруднены унификация, регламентирование и параметризация множества конкретных функций и характеристик сложных объектов архитектуры и структуры современных информационных систем.

Другой подход к разработке и применению профилей информационных систем состоит в использовании совокупности адаптированных и параметризованных базовых международных и национальных стандартов и открытых спецификаций, отвечающих стандартам де-факто и рекомендациям международных консорциумов.

Вопрос 40. Эталонная модель открытых ИС и её составляющие.

Эталонная модель среды открытых систем определяет разделение любой информационной системы на две составляющие: приложения (прикладные программы и программные комплексы) и среду, в которой эти приложения функционируют.

Между приложениями и средой определяются стандартизованные прикладные программные интерфейсы (Application Programming Interface, API), которые являются необходимой частью профилей любой открытой системы. Кроме того, в профилях могут быть определены унифицированные интерфейсы взаимодействия функциональных частей друг с другом и интерфейсы взаимодействия между компонентами среды системы. Спецификации выполняемых функций и интерфейсов взаимодействия могут быть оформлены в виде профилей компонентов системы. Таким образом, профили информационной системы с иерархической структурой могут включать в себя:

1. Стандартизованные описания функций, выполняемых данной системой;

2. Функции взаимодействия системы с внешней для нее средой;

3. Стандартизованные интерфейсы между приложениями и средой информационной системы;

4. Профили отдельных функциональных компонентов, входящих в систему.

Вопрос 41. Использование профилей ИС.

Для эффективного использования конкретного профиля необходимо:

1. Выделить объединенные логической связью проблемно-ориентированные области функционирования, где могут применяться стандарты, общие для одной организации или группы организаций;

2. Идентифицировать стандарты и нормативные документы, варианты их использования и параметры, которые необходимо включить в профиль;

3. Документально зафиксировать части конкретного профиля, в которых требуется создание новых стандартов или нормативных документов, и идентифицировать характеристики, которые могут оказаться важными для разработки недостающих стандартов и нормативных документов этого профиля;

4. Формализовать профиль в соответствии с его категорией, включая стандарты, различные варианты нормативных документов и дополнительные параметры, которые непосредственно связаны с профилем;

5. Опубликовать профиль и/или продвигать его по формальным инстанциям для дальнейшего распространения.

При использовании профилей важно обеспечить корректность их применения путем тестирования, испытаний и сертификации. Для этого

требуется создание технологии контроля и тестирования в процессе применения профиля. Данная технология должна поддерживаться совокупностью методик, инструментальных средств, составом и содержанием оформляемых документов на каждом этапе выполнения проекта.

Использование профилей способствует унификации при разработке тестов, проверяющих качество и взаимодействие компонентов проектируемой информационной системы. Профили должны определяться таким образом, чтобы тестирование их реализации можно было проводить по возможности наиболее полно по стандартизованной методике. При этом возможно применение ранее разработанных методик, так как международные стандарты и профили являются основой для создания общепризнанных аттестационных тестов.

Вопрос 42. Структура профиля ИС.

Разработка и применение профилей являются органической частью процессов проектирования, разработки и сопровождения информационных систем. Профили характеризуют каждую конкретную информационную систему на всех стадиях ее жизненного цикла, задавая согласованный набор базовых стандартов, которым должна соответствовать система и ее компоненты.

Стандарты, важные с точки зрения заказчика, должны задаваться в техническом задании на проектирование системы и составлять ее первичный профиль. То, что не задано в техническом задании, первоначально остается на усмотрение разработчика системы, который, руководствуясь требованиями технического задания, может дополнять и развивать профили системы и впоследствии согласовывать их с заказчиком. Таким образом, профиль конкретной системы не является статичным, он развивается и конкретизируется в процессе проектирования информационной системы и оформляется в составе документации проекта системы.

В профиль конкретной системы включаются спецификации компонентов, разработанных в составе данного проекта, и спецификации использованных готовых программных и аппаратных средств, если эти средства не специфицированы соответствующими стандартами. После завершения проектирования и испытаний системы, в ходе которых проверяется ее соответствие профилю, профиль применяется как основной инструмент сопровождения системы при эксплуатации, модернизации и развитии.

Формирование и применение профилей конкретных информационных систем выполняется на основе международных и национальных стандартов, ведомственных нормативных документов, а также стандартов де-факто при условии доступности соответствующих им спецификаций. Для обеспечения корректного применения профилей их описания должны содержать:

1. Определение целей использования профиля;

2. Точное перечисление функций объекта или процесса стандартизации, определяемого профилем;

3. Формализованные сценарии применения базовых стандартов и спецификаций, включенных в профиль;

4. Сводку требований к информационной системе или к ее компонентам, определяющих их соответствие профилю, и требований к методам тестирования соответствия;

5. Нормативные ссылки на конкретный набор стандартов и других нормативных документов, составляющих профиль, с точным указанием применяемых редакций и ограничений, способных повлиять на достижение корректного взаимодействия объектов стандартизации при использовании профиля;

6. Информационные ссылки на все исходные документы.

При этом структурная модель профиля должна состоять из трех базовых уровней:

1. Архитектурный уровень, определяющий перечень стандартизованных на международном уровне эталонных функциональных моделей, которые должны использоваться при описании информационных систем и среды их исполнения;

2. Функциональный уровень, непосредственно определяющий состав стандартизованных спецификаций;

3. Локальный уровень, состав которого отражается в основном документе по формированию профиля ("Главном профиле") в виде перечня действующих локальных профилей.

Вопрос 43. Основные профили, применяемые на стадиях ЖЦ ИС и их особенности.

На стадиях жизненного цикла информационной системы выбираются и затем применяются следующие основные функциональные профили:

1. Прикладного программного обеспечения;

Прикладное программное обеспечение всегда является проблемно-ориентированным и определяет основные функции информационной системы. Функциональные профили системы должны включать в себя согласованные базовые стандарты. При использовании функциональных профилей информационных систем следует еще иметь в виду согласование этих профилей между собой. Необходимость такого согласования возникает, в частности, при использовании стандартизованных интерфейсов API, в том числе интерфейсов приложений со средой их функционирования и со

средствами защиты информации. При согласовании функциональных профилей возможны также уточнения профиля среды системы и профиля встраиваемых инструментальных средств создания, сопровождения и развития прикладного программного обеспечения.

2. Среды информационной системы;

Профиль среды информационной системы должен определять ее архитектуру в соответствии с выбранной моделью обработки данных.

Стандарты интерфейсов приложений со средой (API) должны быть определены по функциональным областям профилей информационной системы. Декомпозиция структуры среды функционирования системы на составные части, выполняемая на стадии эскизного проектирования, позволяет детализировать профиль среды информационной системы по следующим функциональным областям эталонной модели OSE/RM:

А. Графического пользовательского интерфейса;

Б. Реляционных или объектно-ориентированных СУБД (например, стандарта языка SQL-92 и спецификации доступа к разным базам данных);

В. Операционных систем с учетом сетевых функций, выполняемых на уровне операционной системы;

Г. Телекоммуникационной среды в части услуг и служб прикладного уровня (электронной почты, доступа к удаленным базам данных, передачи файлов, доступа к файлам и управления файлами).

Профиль среды распределенной системы должен включать стандарты протоколов транспортного уровня, стандарты локальных сетей (например, стандарт Ethernet IEEE 802.3 или стандарт Fast Ethernet IEEE 802.3 u), а также стандарты средств сопряжения проектируемой информационной системы с сетями передачи данных общего назначения.

Выбор аппаратных платформ информационной системы связан с определением их параметров: вычислительной мощности серверов и рабочих станций в соответствии с проектными решениями по разделению функций между клиентами и серверами; степени масштабируемости аппаратных платформ; надежности. Профиль среды должен содержать стандарты, определяющие параметры технических средств и способы их измерения (например, стандартные тесты измерения производительности).

3. Защиты информации в информационной системе;

Профиль защиты информации должен обеспечивать реализацию политики информационной безопасности, разрабатываемой в соответствии с требуемой категорией безопасности и критериями безопасности, заданными в техническом задании на систему. Построение профиля защиты информации в распределенных системах клиент-сервер методически связано с точным

определением компонентов системы, ответственных за те или иные функции, службы и услуги, и средств защиты информации, встроенных в эти компоненты. Функциональная область защиты информации включает в себя следующие функции, реализуемые разными компонентами системы:

А. Функции, реализуемые операционной системой;

Б. Функции защиты от несанкционированного доступа, реализуемые на уровне программного обеспечения промежуточного слоя;

В. Функции управления данными, реализуемые СУБД;

Г. Функции защиты программных средств, включая средства защиты от вирусов;

Д. Функции защиты информации при обмене данными в распределенных системах, включая криптографические функции;

Е. Функции администрирования средств безопасности.

Профиль защиты информации должен включать указания на методы и средства обнаружения в применяемых аппаратных и программных средствах не декларированных возможностей. Профиль должен также включать указания на методы и средства резервного копирования информации и восстановления информации при отказах и сбоях аппаратуры системы.

4. Инструментальных средств, встроенных в информационную систему.

Профиль инструментальных средств, встроенных в информационную систему, должен отражать решения по выбору методологии и технологии создания, сопровождения и развития информационной системы. В этом профиле должны содержаться ссылки на описание выбранных методологии и технологии, выполненное на стадии эскизного проектирования системы.

Состав инструментальных средств определяется на основании решений и нормативных документов об организации сопровождения и развития информационной системы. При этом должны быть учтены правила и порядок, регламентирующие внесение изменений в действующие системы. Функциональная область профиля инструментальных средств, встроенных в систему, охватывает функции централизованного управления и администрирования, связанные с:

А. Контролем производительности и корректности функционирования системы в целом;

Б. Конфигурированием прикладного программного обеспечения, тиражированием версий;

В. Управлением доступом пользователей к ресурсам системы и конфигурированием ресурсов;

Г. Перенастройкой приложений в связи с изменениями прикладных функций информационной системы;

Д. Настройкой пользовательских интерфейсов (экранных форм и отчетов);

Е. Ведением баз данных системы;

Ж. Восстановлением работоспособности системы после сбоев и аварий.

Дополнительные ресурсы, необходимые для функционирования встроенных инструментальных средств, такие как минимальный и рекомендуемый объемы оперативной памяти, размеры требуемого дискового пространства и т. п., должны быть учтены в разделе проекта, относящемся к среде информационной системы.

Выбор инструментальных средств, встроенных в систему, должен производиться в соответствии с требованиями профиля среды. Ссылки на соответствующие стандарты, входящие в профиль среды, должны содержаться и в профиле инструментальных средств.

В этом профиле должны также содержаться ссылки на требования к средствам тестирования, которые необходимы для сопровождения и развития системы и должны быть в нее встроены. В число встроенных в информационную систему средств тестирования должны входить средства функционального тестирования приложений, тестирования интерфейсов, системного тестирования и тестирования серверов/клиентов при максимальной нагрузке.

Вопрос 44. Процесс управления конфигурацией информационных систем.

Управление конфигурацией – один из вспомогательных процессов, поддерживающих основные процессы жизненного цикла ПО, прежде всего процессы разработки и сопровождения ПО ИС.

Каждый процесс характеризуется определёнными задачами и методами их решения, исходными данными, полученными на предыдущем этапе, и результатами. Результатами анализа, в частности, являются функциональные модели, информационные модели и соответствующие им диаграммы. ЖЦ ПО носит итерационный характер: результаты очередного этапа часто вызывают изменения в проектных решениях, выработанных на более ранних этапах.

К управлению конфигурацией следует отнести функции анализа производительности и оптимизации системы.

Большинство систем имеют оптимальные настройки по умолчанию и не требуют особого вмешательства. Однако, производители сетевых операционных систем включают в них наборы эмпирических правил, помогающих администратору вносить изменения в настройки с минимальным риском ухудшить другие показатели или сделать систему неработоспособной. Администратору следует их изучить и знать перечень параметров, которые необходимо контролировать. Многих проблем можно избежать еще на стадии планирования сети.

Сюда же можно отнести задачу, связанную с учётом системных ресурсов. Учёт ресурсов позволяет заметить тенденции к появлению узких мест до того, как появятся проблемы с производительностью и провести соответствующую модернизацию. Кроме того, система учёта необходима 80 при платном использовании ресурсов, например, контроль использования дискового пространства, печати, учёт трафика.

Вопрос 45. Свойство управляемости сети.

Управляемость сети подразумевает возможность централизованно контролировать состояние основных элементов сети, выявлять и разрешать проблемы, возникающие при работе сети, выполнять анализ производительности и планировать развитие сети. В идеале средства управления сетями представляют систему, осуществляющую наблюдение, контроль и управление каждым элементом сети: от простейших до самых сложных устройств. При этом такая система рассматривает сеть как единое целое, а не как разрозненный набор отдельных устройств.

Хорошая система управления наблюдает за сетью и, обнаружив проблему, активизирует определённое действие, исправляет ситуацию и уведомляет администратора о том, что произошло и какие шаги предприняты. Одновременно с этим система управления должна накапливать данные, на основании которых можно планировать развитие сети. Система управления должна быть независимой от производителя и обладать удобным интерфейсом, позволяющим выполнять все действия с одной консоли.

Полезность системы управления особенно ярко проявляется в больших сетях: корпоративных или публичных глобальных. Без неё в таких сетях нужно присутствие квалифицированных специалистов по эксплуатации в каждом здании каждого города, где установлено оборудование сети, что в итоге приводит к необходимости содержания 81 огромного штата обслуживающего персонала. Большинство существующих средств не управляют сетью, а осуществляют наблюдение за её работой. Они следят за сетью, но не выполняют активных действий, если с сетью что-то произошло или может произойти. Мало масштабируемых систем, способных обслуживать как сети масштаба отдела, так и сети масштаба предприятия.

Большинство систем управляют отдельными элементами сети и не анализируют способность сети выполнять качественную передачу данных между конечными её пользователями.

Вопрос 46. Свойство совместимости.

Совместимость или интегрируемость означает, что сеть способна включать в себя самое разнообразное программное и аппаратное обеспечение, то есть в ней могут сосуществовать различные операционные системы, поддерживающие разные стеки коммуникационных протоколов, и работать аппаратные средства и приложения от разных производителей.

Сеть состоит из огромного числа различных модулей – компьютеров, сетевых адаптеров, мостов, маршрутизаторов, модемов, операционных систем и модулей приложений. Не существует компании, которая смогла 82 бы обеспечить производство полного набора всех типов и подтипов оборудования и программного обеспечения, требуемого для построения сети.

Вопрос 47. Свойство модульности и расширяемости.

Модульность – одно из неотъемлемых и естественных свойств вычислительных сетей. Модульность отражается не только в многоуровневом представлении коммуникационных протоколов в конечных узлах сети – важная и принципиальная особенность сетевой архитектуры.

Все компоненты сети должны работать согласованно, для этого оказалось необходимым принять много стандартов, которые гарантировали бы совместимость оборудования и программ различных фирм-изготовителей. Таким образом, понятия модульности и стандартизации в сетях неразрывно связаны, и модульный подход только тогда даёт преимущества, когда он сопровождается следованием стандартам.

Расширяемость означает возможность сравнительно легкого добавления отдельных элементов сети (пользователей, компьютеров, приложений, служб), наращивания длины сегментов сети и 83 замены существующей аппаратуры более мощной. При этом принципиально важно, что легкость расширения системы иногда может обеспечиваться в некоторых весьма ограниченных пределах. Например, локальная сеть Ethernet, построенная на основе одного сегмента толстого коаксиального кабеля, обладает хорошей расширяемостью, в том смысле, что позволяет легко подключать новые станции.

Однако такая сеть имеет ограничение на число станций – их число не должно превышать 30-40. Хотя сеть допускает физическое подключение к сегменту и большего числа станций (до 100), но при этом чаще всего резко снижается производительность сети. Наличие такого ограничения и является признаком плохой масштабируемости системы при хорошей расширяемости.

Вопрос 48. Свойство масштабируемости.

Масштабируемость означает, что сеть позволяет наращивать количество узлов и протяжённость связей в очень широких пределах, при

этом производительность сети не ухудшается. Для обеспечения масштабируемости сети приходится применять дополнительное коммуникационное оборудование и специальным образом структурировать сеть. Например, хорошей масштабируемостью обладает многосегментная сеть, построенная с использованием коммутаторов и маршрутизаторов и имеющая иерархическую структуру связей. Такая сеть может включать несколько тысяч компьютеров и при этом обеспечивать каждому пользователю сети нужное качество обслуживания.

Вопрос 49. Аудит информационных систем. Понятие «информационный аудит» и причины его проведения.

Информационный аудит – это проверка и оценка практики использования ИТ-систем в организации, осуществляемая специализированной независимой организацией. Выделяют две причины необходимости проведения такого аудита.

Первая причина – информационным технологиям большинства Российских компаний свойственен эволюционный путь создания и дальнейшего их развития. Он характеризуется тем, что информационные системы включаются в инфраструктуру ИТ или модернизируются по мере возникновения необходимости и (или) по мере возможности (в том числе и финансовой). В итоге в ИТ-инфраструктуре формируется сложная (порой разнородная) и поэтому плохо управляемая совокупность программно-технических и системных платформ. Даже если ИС прошла все стадии создания, последующие изменения бизнес-процессов или введение новых приложений могут привести к тому, что параметры её программно-аппаратных платформ перестанут соответствовать требованиям бизнеса.

Вторая причина связана с зависимостью успешности бизнеса от способности управленцев вовремя получать и быстро обрабатывать 86 нужную информацию. Очевидно, что современный руководитель любого звена не в состоянии одновременно быть компетентным в различных областях, обеспечивать получение и обработку разноплановой информации.

Вопрос 50. Три части системы аудита ИС.

- внутренний контроль, осуществляемый организацией, эксплуатирующей ИС;
- внешний контроль, осуществляемый внешней организацией в рамках разделения полномочий организации, эксплуатирующей ИС, и вышестоящего ведомства или организации;
- независимый информационный аудит.

Вопрос 51. Виды аудита ИС.

- процессов управления службой ИТ,

- структуры службы ИТ,
- информационной системы,
- ТЗ и проектной документации на создание ИС,
- систем резервирования данных.

Вопрос 52. Свойства ИС, оцениваемые аудитом.

Аудит информационной системы используется для определения (оценки степени) соответствия сервисов текущим и планируемым в будущем потребностям бизнеса организации и услугам, предоставляемым информационной системой сотрудникам организации.

Результатом такого аудита являются предварительные рекомендации по совершенствованию (модернизации) и предложения по структуре проекта совершенствования (модернизации) ИС.

Вопрос 53. Тестирование информационных систем. Понятие и критерии.

Под тестированием понимается процесс исполнения программы с целью обнаружения ошибок.

Регрессионное тестирование – это тестирование, проводимое после усовершенствования функций программы или внесения в неё изменений. Одно из средств тестирования QA (ныне – Quality Works) представляет интегрированную, многоплатформенную среду разработки автоматизированных тестов любого уровня, включая тесты регрессии для приложений с графическим интерфейсом пользователя.

Критерии тестирования включают:

● описание тестов;

● фиксацию повторения действий оператора (возможность фиксировать данные, вводимые оператором с помощью клавиатуры, 90 мыши и т.д., редактировать их и воспроизводить в тестовых примерах);

● автоматический запуск тестовых примеров;

● регрессионное тестирование (возможность повторения и модификации ранее выполненных тестов для определения различий в системе и/или среде);

● автоматизированный анализ результатов тестирования и исключительных ситуаций в процессе тестирования, включая сравнение ожидаемых и реальных результатов, сравнение файлов, статистический анализ результатов; обращения к операторам, процедурам и переменным; защиту от несанкционированного доступа и др.;

● анализ производительности. Анализируемые параметры производительности могут включать использование центрального процессора, памяти, обращения к определённым элементам данных и (или) сегментам кода, временные характеристики и т.д.

Вопрос 54. Результаты оценки тестируемой системы. Отчет по результатам оценки.

Результаты оценки должны быть стандартным образом документированы и, при необходимости, утверждены. Отчёт по результатам оценки должен содержать следующую информацию:

● введение (общий обзор процесса и перечень основных результатов);

● предпосылки (цель оценки и желаемые результаты, период времени, в течение которого выполнялась оценка, определение ролей и соответствующего опыта специалистов, выполнявших оценку);

● подход к оценке (описание общего подхода, включая информацию, определяющую контекст и масштаб оценки, а также любые 91 предположения и ограничения);

● информация об ИС, которая должна включать:

наименование системы;

данные о Заказчике и Исполнителях, включая контактную информацию;

конфигурацию технических средств;

стоимостные данные;

описание ИС, включающее поддерживаемые данным средством процессы создания и сопровождения ИС, программную среду (поддерживаемые языки программирования, операционные системы, совместимость с базами данных), функции, входные/выходные данные и область применения;

● этапы оценки (конкретные действия, выполняемые в процессе оценки, должны быть описаны со степенью детализации, необходимой как для понимания масштаба и глубины оценки, так и для её повторения при необходимости);

● конкретные результаты должны быть представлены в терминах критериев оценки. В тех случаях, когда отчёт охватывает ряд средств или результаты данной оценки будут сопоставляться с аналогичными результатами других оценок, необходимо обратить особое внимание на формат представления результатов, способствующий такому сравнению. Субъективные результаты отделяют от объективных и сопровождают необходимыми пояснениями;

● выводы и заключения;

● приложения, в которых присутствуют формулировка задачи оценки 92 и уточненный список критериев. В процессе реализации проекта важное место занимают вопросы идентификации, описания и контроля конфигурации отдельных компонентов и всей системы в целом.

Вопрос 55. Эксплуатация и техническое обслуживание ИС.

Эксплуатация включает работы по внедрению компонентов ПО в эксплуатацию, в том числе конфигурирование БД и рабочих мест пользователей, обеспечение эксплуатационной документацией, проведение обучения персонала и т.д., и непосредственно эксплуатацию, в том числе локализацию проблем и устранение причин их возникновения, модификацию ПО в рамках установленного регламента, подготовку предложений по совершенствованию, развитию и модернизации системы.

Продуктивность и иные характеристики деятельности организации после внедрения в ней разработанной ИС могут первоначально ухудшиться, так как на освоение новых средств и внесение необходимых изменений в процессы разработки и эксплуатации требуется некоторое время. Таким образом, ожидаемые результаты должны рассматриваться с учётом вероятной отсрочки в улучшении проектных и эксплуатационных характеристик.

Техническое обслуживание и модернизация. Если собственно техническое обслуживание (очистка от пыли, смазка вентиляторов, подтяжка креплений, контроль состояния аккумуляторов, изменение физической топологии сети и т. п.) может осуществляться службой 93 технической поддержки, то грамотное формулирование заявок на изменение аппаратной конфигурации, организация закупки дополнительных лицензий или обновленной версии программного обеспечения – задача администратора.

Вопрос 63. Принципы и регламентация документирования.

В части оформления проектной документации автоматизированных систем на сегодняшний день существует достаточно приемлемая, хотя и нуждающаяся в совершенствовании регламентация, а именно группа ГОСТов, относящихся к 34 серии, в частности:

ГОСТ 34.201-89 Информационная технология. Комплекс стандартов на автоматизированные системы. Виды, комплектность и обозначения документов при создании автоматизированных систем.

ГОСТ 34.601-90 Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания.

ГОСТ 34.602-89 Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы

ГОСТ 34.603-92 Информационная технология. Виды испытаний автоматизированных систем

РД 50-34.698-90 Автоматизированные системы. Требования к содержанию документов.

Вопрос 64. Рекомендации документирования согласно ГОСТам.

Указанные нормативно-технические документы позволяют достаточно полно и качественно описать разработанную информационную систему и обосновать принятые в ней проектные решения – при условии жесткого соблюдения исполнителем требований состава, назначения и структуры документов. При этом рекомендуется 104 дополнительно ограничивать предусмотренную в ГОСТ возможность исключения и слияния подразделов документации, что часто используется исполнителями для произвольного толкования содержательных требований стандарта. Аналогичная проблема возникает в связи с тем, что состав документов, предусмотренный ГОСТ 34.201-89, является в значительной степени избыточным, а часть из них либо утратила смысл в силу развития технологий, либо не применима к большинству информационных систем (например, «Схема соединений» для проектов, в рамках которых не производится поставка технических средств, или «Пакеты входных/выходных данных» для систем, реализующих взаимодействие). С другой стороны, документация стадий эскизного и технического проектировании имеет каскадную структуру – большинство документов соответствуют определенным разделам общей пояснительной записке, дополняя или уточняя их. В результате простую общую ссылку на ГОСТ 34.201-89 в условиях госконтракта следует считать недостаточной. При предоставлении исполнителю права самостоятельно определять состав необходимых проектных и рабочих документов зачастую возникает ситуация либо избыточного документирования (в проект включаются все предусмотренные документы, в т.ч. бессмысленные для данного проекта или дублирующие друг друга), либо недостаточного (все решения по проекту описываются в одной пояснительной записке, что перегружает документ или не позволяет представить всю необходимую информацию). Также на стадии «РД» зачастую опускается документ «Общее описание системы», что не позволяет оценить, как именно были реализованы решения стадий ЭП/ТП. 105 С учетом рекомендаций, изложенных в разделе Представление программного обеспечения требование об обязательной разработке общего описания следует прямо включать в условия госконтракта (в задание или рабочую программу), а также определять точный состав других документов, подлежащих разработке. Кроме того, рекомендуется более четко выполнять привязку календарных планов госконтрактов к стадийности разработки АС по ГОСТ 34.601. Состав этапов в таблице ниже дан для справки, он может изменяться и дополняться в зависимости от потребностей процесса разработки. Аналогично из проектов могут исключаться отдельные стадии, например, эскизного проектирования. При этом необходимым является соблюдение правильного именования и последовательности стадий, что позволит корректно трактовать положения прочих требований ГОСТ по документированию.