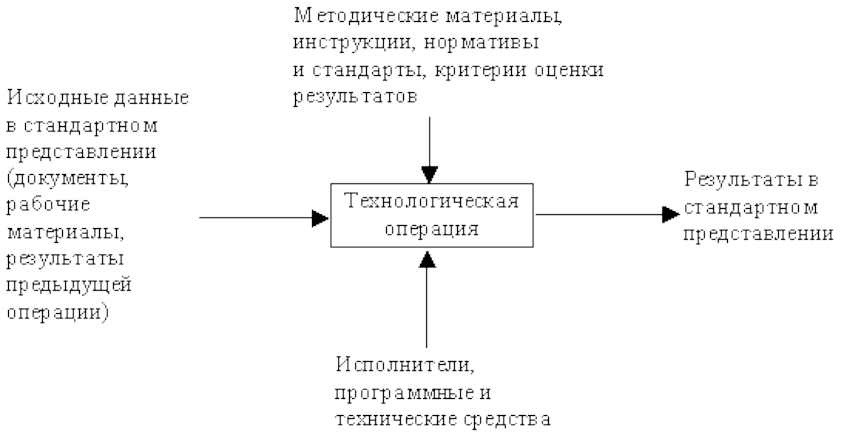
Тема. МЕТОДОЛОГИЯ РАЗРАБОТКИ ИНФОРМАЦИОННЫХ СИСТЕМ

1. Методы проектирования

Методы проектирования ИС подразумевают использование определённых программных и аппаратных средств, составляющих *инструментальные средства программирования ИС*.

Метод проектирования включает совокупность трёх составляющих:

1) пошаговой процедуры, определяющей последовательность технологических операций проектирования;



2) критериев и правил, используемых для оценки результатов выполнения технологических операций;

3) нотаций (графических и текстовых средств), используемых для описания проектируемой системы.

Технологическую операцию считают элементарным (простым) технологическим процессом. При этом *информационная операция* – это отдельная законченная часть процесса (изменение содержания областей смыслового пространства субъекта) или инструкция.

Выделяют *три основных вида проектирования объектов и систем* по степени их сложности, объёму и ряду других показателей:

- *крупные проекты* - при их реализации обычно прибегают к помощи хорошо зарекомендовавших себя крупных компаний-интеграторов, в том числе консалтинговых и внедренческих организаций.
- *средние проекты* - для их реализации стараются обойтись своими силами и (или) используют готовые решения, которые стремятся адаптировать под конкретные требования организации-заказчика.
- *малые (мелкие) проекты* - характеризуются использованием готовых решений и, в ряде случаев, адаптацией их под конкретные условия использования.

Организация проектирования предполагает определение методов взаимодействия проектировщиков между собой и с заказчиком в процессе создания проекта ИС.

Методы проектирования АИС классифицируют по 3 признакам:

По степени автоматизации различают:

- *ручное проектирование*, при котором проектирование компонентов АИС осуществляется без использования специальных инструментальных программных средств; программирование производится на алгоритмических языках;
- *компьютерное проектирование*, при котором генерация или конфигурация (настройка) проектных решений производится с использованием специальных инструментальных программных средств.

По степени использования типовых проектных решений различают:

- *оригинальное (индивидуальное)* проектирование, когда проектные решения разрабатываются «с нуля» в соответствии с требованиями к АИС; *Оригинальное проектирование* АИС предполагает максимальный учет особенностей автоматизированного объекта.
- *типовое проектирование*, предполагающее конфигурацию АИС из готовых типовых проектных решений (программных модулей). *Типовое проектирование* выполняется на основе готовых решений и является обобщением опыта, полученного ранее при создании родственных проектов.

По степени адаптивности проектных решений различаются следующие методы:

- *реконструкция* — адаптация проектных решений выполняется путем переработки соответствующих компонентов (перепрограммирования программных модулей);
- *параметризация* — проектные решения настраиваются в соответствии с заданными и изменяемыми параметрами;
- *реструктуризация модели* — изменяется модель предметной области, что приводит к автоматическому переформированию проектных решений.

Осуществление проектирования ИС предполагает использование проектировщиками определенной технологии проектирования, соответствующей масштабу и особенностям разрабатываемого проекта.

Технология проектирования АИС — это совокупность методов и средств проектирования АИС, а также методов и средств организации проектирования (управление процессом создания и модернизации проекта АИС).

Технология проектирования АИС реализует определенную методологию проектирования.

Методология проектирования предполагает наличие некоторой концепции, принципов проектирования и реализуется набором методов и средств.

Современные методологии проектирования систем должны обеспечивать описание объектов автоматизации, описание функциональных возможностей АИС, спецификацию проекта, гарантирующую достижение заданных характеристик системы, детальный план создания системы с оценкой сроков разработки, описание реализации конкретной системы.

2. Основные методологии разработки информационных систем

Методологии внедрения информационных систем являются источником информации для разработки иерархической структуры проекта внедрения и иерархической структуры работ проекта. Состав работ (процессов) и последовательность их исполнения в значительной мере определяются целями проекта внедрения, используемым программным обеспечением, особенностями автоматизируемой сферы деятельности, организационной структурой объекта автоматизации, принятой у разработчика организацией работы и пр.

MICROSOFT SOLUTIONS FRAMEWORK (MSF) - это методология ведения проектов и разработки решений, базирующаяся на принципах работы над продуктами самой фирмы Microsoft и предназначенная для использования в организациях, нуждающихся в концептуальной схеме для построения современных решений.

MSF опирается на практический опыт Microsoft и описывает управление людьми и рабочими процессами в процессе разработки решения.

Методология Microsoft Solutions Framework (MSF) носит универсальный характер и может использоваться для внедрения произвольной разрабатываемой в ходе проекта системы. Она может быть применена при разработке весьма широкого круга систем: традиционного программного

обеспечения, *ERP*-систем, решений в области *электронного бизнеса*, распределенных сетевых приложений и пр.

Microsoft Solutions Framework является схемой для принятия решений по планированию и реализации новых технологий в организациях. MSF включает обучение, информацию, рекомендации и инструменты для идентификации и структуризации информационных потоков бизнес-процессов и всей информационной инфраструктуры новых технологий.

MSF состоит из двух моделей и трех дисциплин:

Модель процессов MSF. Представляет собой общую методологию разработки и внедрения IT-решения, охватывающую весь жизненный цикл создания решения. Эта модель сочетает в себе особенности двух классических моделей процессов: спиральной и каскадной.

Модель проектной группы MSF. Одной из основ MSF является постулат о шести качественных целях, достижение которых определяет успешность проекта. Исходя из этих целей построена модель проектной группы. Она включает в себя шесть ролевых кластеров, каждый из которых имеет свою область компетенции, а также связанные с нею цели и задачи.

Дисциплина управления проектами. Данная дисциплина описывает основные принципы, концепции и характеристики управления проектом в рамках MSF. Она отвечает вопрос: Что такое управление проектом?

Дисциплина управления рисками. MSF рассматривает изменения и связанные с ними затруднения как неотъемлемую часть жизненного цикла информационных технологий. MSF отстаивает превентивный подход к работе с рисками. Должна осуществляться непрерывная оценка рисков на протяжении всего жизненного цикла проекта. Данная дисциплина предлагает принципы, идеи и рекомендации, подкрепленные описанием пошагового процесса для успешного активного управления рисками. Этот процесс включает в себя выявление и анализ рисков; планирование и реализацию стратегий по их профилактике и смягчению возможных последствий; отслеживание состояния рисков и извлечение уроков из обретенного опыта.

Дисциплина управления подготовкой. Данная дисциплина посвящена управлению знаниями, профессиональными умениями и способностями, необходимыми для планирования, создания и сопровождения успешных решений. Дисциплина управления подготовкой MSF описывает фундаментальные принципы MSF и дает рекомендации по применению превентивного подхода к управлению знаниями на протяжении всего жизненного цикла информационных технологий. Эта дисциплина также рассматривает планирование процесса управления подготовкой. Будучи подкрепленной испытанными практическими методиками, дисциплина управления подготовкой предоставляет проектным группам и отдельным специалистам базу для осуществления этого процесса.

В MSF управление разработкой распределено между руководителями отдельных проектных групп внутри коллектива, выполняющих следующие задачи:

- Управление программой
- Разработка
- Тестирование
- Управление выпуском
- Удовлетворение потребителя
- Управление продуктом

Модель процессов MSF отражает интегрированную (общую) *методологию разработки* и внедрения **ИТ-решений.**

Под ИТ-решением в MSF понимается скоординированная поставка набора элементов (таких как программно-технические средства, документация, обучение и сопровождение), необходимых для удовлетворения некоторой бизнес потребности конкретного заказчика. Основными компонентами решения являются:

- программно-технические средства, которые могут быть как новыми, так и усовершенствованными версиями разработанных ранее;
- внедрение - включает в себя процедуры установки/удаления аппаратного и программного обеспечения;

- обучение - процедуры, которые распространяются на всех участников использования и сопровождения решения;
- документация - вся информация, необходимая для установки, поддержки, сопровождения и использования решения;
- сопровождение - процедуры развития, восстановления, действий в нештатных ситуациях и поддержки пользователей;
- внешние коммуникации - информирование заинтересованных сторон о ходе внедрения решения и его влиянии на их интересы.

Жизненный цикл процессов в MSF сочетает каскадную и *спиральную модели* разработки: проект реализуется поэтапно, с наличием соответствующих контрольных точек, а сама последовательность этапов может повторяться по спирали. Благодаря промежуточным контрольным точкам и обратной спирали верификации облегчается взаимодействие с заказчиком.

Рис. Модель жизненного цикла решения MSF

Фазы проекта определяют последовательно решаемые задачи, а вехи - ключевые точки проекта, характеризующие достижение какого-либо существенного результата.

При управлении проектом четко ставится цель, которую необходимо достичь в результате, и учитываются ограничения, накладываемые на проект. Все виды ограничений могут быть отнесены к одному из трех видов: ограничения ресурсов, ограничения времени и ограничения возможностей. Эти три вида ограничений и приоритетность задач по их преодолению образуют *треугольник приоритетов* в MSF.

Рис. Треугольник приоритетов в MSF

Microsoft выпустила среду разработки, в полной мере поддерживающей основные идеи MSF - Microsoft Visual Studio 2005 Team Edition. Это первый программный комплекс, представляющий собой не среду разработки для индивидуальных членов коллектива, а комплексное средство поддержки коллективной работы.

RATIONAL UNIFIED PROCESS (RUP)- это методология создания программного обеспечения, оформленная в виде размещаемой на Web базы знаний, которая снабжена поисковой системой.

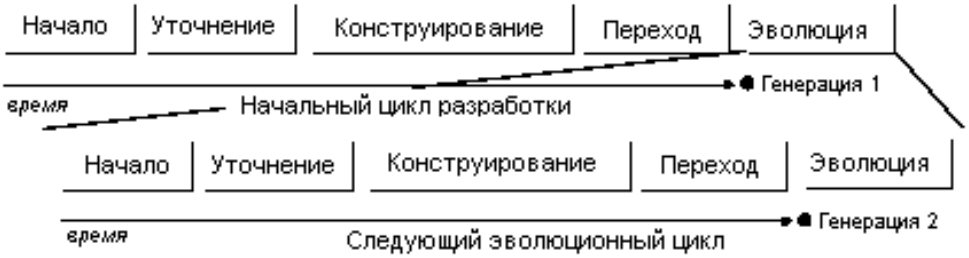
Продукт Rational Unified Process (RUP) разработан и поддерживается Rational Software. Он регулярно обновляется с целью учета передового опыта и улучшается за счет проверенных на практике результатов.

Rational Unified Process – это итеративный процесс (так называемая спиральная модель жизненного цикла разработки). Каждая итерация может приводить к созданию фрагмента разрабатываемой системы или новой версии и включает этапы выработки требований, анализа, проектирования, реализации и тестирования. Поскольку тестирование проводится на каждой итерации, риск снижается уже на начальных этапах жизненного цикла разработки.

Rational Unified Process делит один цикл развития на четыре последовательных стадии:

- Начало
- Уточнение
- Конструирование
- Переход

Каждая стадия заканчивается четко определенной вехой – временной точкой, в которой должны быть принятыми некоторые проблемные решения, а поэтому ключевые цели должны быть
достигнуты.



Стадии и главные вехи процесса.

Первый цикл выполнения этих четырех стадий для данного изделия называют начальным циклом разработки. Если жизненный цикл изделия на этом не завершается, существующее изделие разовьется в своем следующем поколении, повторив ту же последовательность стадий начала, уточнения, конструирования и перехода. Этот период называется "эволюцией". Циклы развития, которые следуют за начальным циклом развития, называются "эволюционными циклами".

Начальная стадия - На начальной стадии разработчик устанавливает деловые применения системы и определяет рамки проекта. Чтобы сделать это, нужно идентифицировать все внешние объекты, с которыми взаимодействует система (субъекты), и определить характер этого взаимодействия на высоком уровне. Эта работа включает идентификацию всех прецедентов и описание нескольких наиболее существенных. Деловое применение включает критерии успеха, оценку риска, оценку необходимых ресурсов и укрупненный план с указанием дат завершения главных этапов.

В конце начальной стадии разработчик исследует требования жизненного цикла проекта и решает, следует ли продолжать разработку.

Стадия уточнения - Цели стадии уточнения состоят в том, чтобы проанализировать прикладную область, создать нормальную архитектурную основу, разработать план проекта и устранить самые высокие элементы риска проекта. Архитектурные решения должны приниматься с пониманием целостной системы. Это подразумевает описание большинства прецедентов и рассмотрение дополнительных требований. Чтобы проверить архитектуру, нужно разработать систему, которая демонстрирует архитектурные решения и выполняет существенный прецедент.
В конце стадии уточнения разработчик детально исследует цели и контекст системы, архитектурные решения и способы разрешения главных рисков.

Стадия конструирования - В ходе стадии конструирования происходит итеративная разработка законченного изделия, которое готово к передаче его пользователям. Это подразумевает описание оставшихся прецедентов, изложение деталей конструкции, завершение выполнения и проверку программного обеспечения.
В конце стадии конструирования разработчик решает, все ли программное обеспечение, рабочие места и пользователи готовы и работоспособны.

Переходная стадия - В процессе переходной стадии разработчик передает программное обеспечение его пользователям. Как только изделие попадает в руки пользователей, часто возникают новые проблемы, которые требуют дополнительной разработки для корректировки системы, исправлению не обнаруженных ранее проблем или завершению реализации некоторых возможностей, которые, возможно, были отложены. Эта стадия обычно начинается с выпуска "бета-версии" системы.

В конце переходной стадии разработчик решает, были ли достигнуты цели жизненного цикла, и возможно, запускает другой цикл разработки. Это также та точка, в которой разработчик может проанализировать результаты и сделать выводы из некоторых уроков, полученных в процессе разработки проекта.

RUP обеспечивает строгий подход к распределению задач и ответственности внутри организации-разработчика. Его предназначение заключается в том, чтобы гарантировать создание точно в срок и в рамках установленного бюджета качественного ПО, отвечающего нуждам конечных пользователей.

RUP способствует повышению производительности коллективной разработки и предоставляет лучшее из накопленного опыта по созданию ПО, посредством руководств, шаблонов и наставлений по пользованию инструментальными средствами для всех критически важных работ, в течение жизненного цикла создания и сопровождения ПО. Обеспечивая каждому члену группы доступ к той же самой базе знаний, вне зависимости от того, разрабатывает ли он требования, проектирует, выполняет тестирование или управляет проектом - RUP гарантирует, что все члены группы используют общий язык моделирования и процесс, имеют согласованное видение того, как создавать ПО. В качестве языка моделирования в общей базе знаний используется Unified Modeling Language (UML), являющийся международным стандартом.

RUP - это конфигурируемый процесс, RUP может конфигурироваться для учета различных ситуаций. В его состав входит Development Kit, который обеспечивает поддержку процесса конфигурирования под нужды конкретных организаций. RUP пригоден как для маленьких групп разработчиков, так и для больших организаций, занимающихся созданием ПО.

RUP представлен в виде “on-line” документации, оформленной как web-страницы, что позволяет размещать его описание на сервере внутренней сети предприятия с целью приобщения всех сотрудников к объектно-ориентированной методологии.