

Разработка
требований и
внешнее
проектирование ПО

Содержание

- 0 Общая схема процесса создания ПО
- 0 Разработка требований к ПО
- 0 Цели разработки ПО
- 0 Разработка внешних спецификаций проекта
- 0 Технологии проектирования ПО

Общая схема процесса создания ПО

Постановка
задач



Алгоритми-
зация
решения
задач



Программи-
рование

- 0 **Постановка задачи** (*problem definition*) – это точная формулировка решения задачи на компьютере с описанием входной и выходной информации.
- 0 **Алгоритм** – система точно сформулированных правил, определяющая процесс преобразования допустимых исходных данных (входной информации) в желаемый результат (выходную информацию) за конечное число шагов.
- 0 **Программирование** (*programming*) – теоретическая и практическая деятельность, связанная с созданием программ.
- 0 **Программа** – алгоритм решения задачи, формализованный на языке программирования.

Основные характеристики функциональных задач в процессе ее формализованной постановки

- 0 цель или назначение задачи, ее место и связи с другими задачами;
- 0 -условия решения задачи с использованием средств вычислительной техники;
- 0 -содержание функций обработки входной информации при решении задачи;
- 0 -требования к периодичности решения задачи;
- 0 -ограничения по срокам и точности выходной информации;
- 0 -состав и форма представления выходной информации;
- 0 -источники входной информации для решения задачи;
- 0 -пользователи задачи (кто осуществляет ее решение и пользуется результатами решения и пользуется результатами решения).

Алгоритм решения задачи имеет ряд обязательных свойств:

- 0 -дискретность – разбиение процесса обработки информации на более простые этапы (шаги), выполнение которых компьютером или человеком не вызывая затруднений;
- 0 -определённость алгоритма – однозначность выполнения каждого отдельного шага преобразования информации;
- 0 -выполнимость – конечность действий алгоритма решения задач, позволяющая получить желаемый результат при допустимых исходных данных за конечное число шагов;
- 0 -массовость – пригодность алгоритма для решения определенного класса задач.

Перевод на промышленную основу

- 0 - стандартизованность, тиражируемость и воспроизведение различными разработчиками методов программирования;
- 0 - внедрение прогрессивных инструментальных средств разработки программ;
- 0 - использование специальных методов и приемов организации работ по раз-работке программ.

- Прикладной программист
- Постановщики задач
- Программист-аналитик
- Системный программист

Разработка требований к ПО

Три группы программных проектов:

- управляемые пользователем,
- утверждаемые пользователем
- независимые от пользователя.

Фазы в выработке требований

- Фаза планирования

- определяется реализуемость, устанавливаются цели, оцениваются затраты и обеспечивается ориентация для разработки проекта

- Фаза выработки

- вырабатываются требования к входным данным, информационным потокам, выходным данным, документации, среде, вычислительным ресурсам

Результатом работы по выработке требований обычно является соответствующий документ

- 0** • достаточным для идентификации целей ПИ, его среды, преимуществ и недостатков ПИ для пользователя, состава и конфигурации ресурсов для его работы;
- 0** • достаточно полным, чтобы в последующем при разработке исключить серьезные модификации и пересмотр требований;
- 0** • достаточным для просмотра и утверждения администрацией на основе его реализуемости в соответствии с выбранными критериями.

Цели разработки ПО

1. Краткое описание.
2. Определение пользователя.
3. Подробное описание функциональных задач.
4. Документация.
5. Эффективность.
6. Совместимость.
7. Конфигурация.
8. Безопасность.
9. Обслуживание.
10. Установка.
11. Надежность.

Надежность

- 0 среднее время наработки на сбой для каждого вида сбоя (ПИ, пользователь, отдельная функция) и степень важности сбоя;**
- 0 среднее время восстановления ПИ после сбоя;**
- 0 цели по числу ошибок ПИ по категориям сложности и времени обнаружения;**
- 0 последствия сбоев системы и наиболее важных функций;**
- 0 допустимый объем данных, утрачиваемых во время сбоя, и уровень обеспечения безопасности;**
- 0 функции, необходимые для обнаружения и исправления ошибок, а также обеспечение устойчивости к ним;**
- 0 возможности обнаружения ошибок пользователя и аппаратуры, а также восстановления работоспособности.**

Разработка внешних спецификаций проекта

Внешнее проектирование — это процесс описания планируемого поведения разрабатываемого ПИ с точки зрения потенциальных пользователей.

Целью этого процесса является конкретизация внешних взаимодействий будущего ПИ без детализации внутреннего устройства.

0 Система с **концептуальной целостностью** должна иметь следующую характеристику:

«все средства, доступные одному пользователю, должны быть доступны и другим пользователям, т.е. интерфейсы пользователей должны быть идентичны.»

Проектировщик должен решить три проблемы:

- 0 1) доведение до минимума ошибок пользователя;
- 0 2) обнаружение ошибок пользователя в случае их возникновения;
- 0 3) доведение до минимума сложности разрабатываемого ПИ.

Разработка внешних спецификаций разбивается на две части:

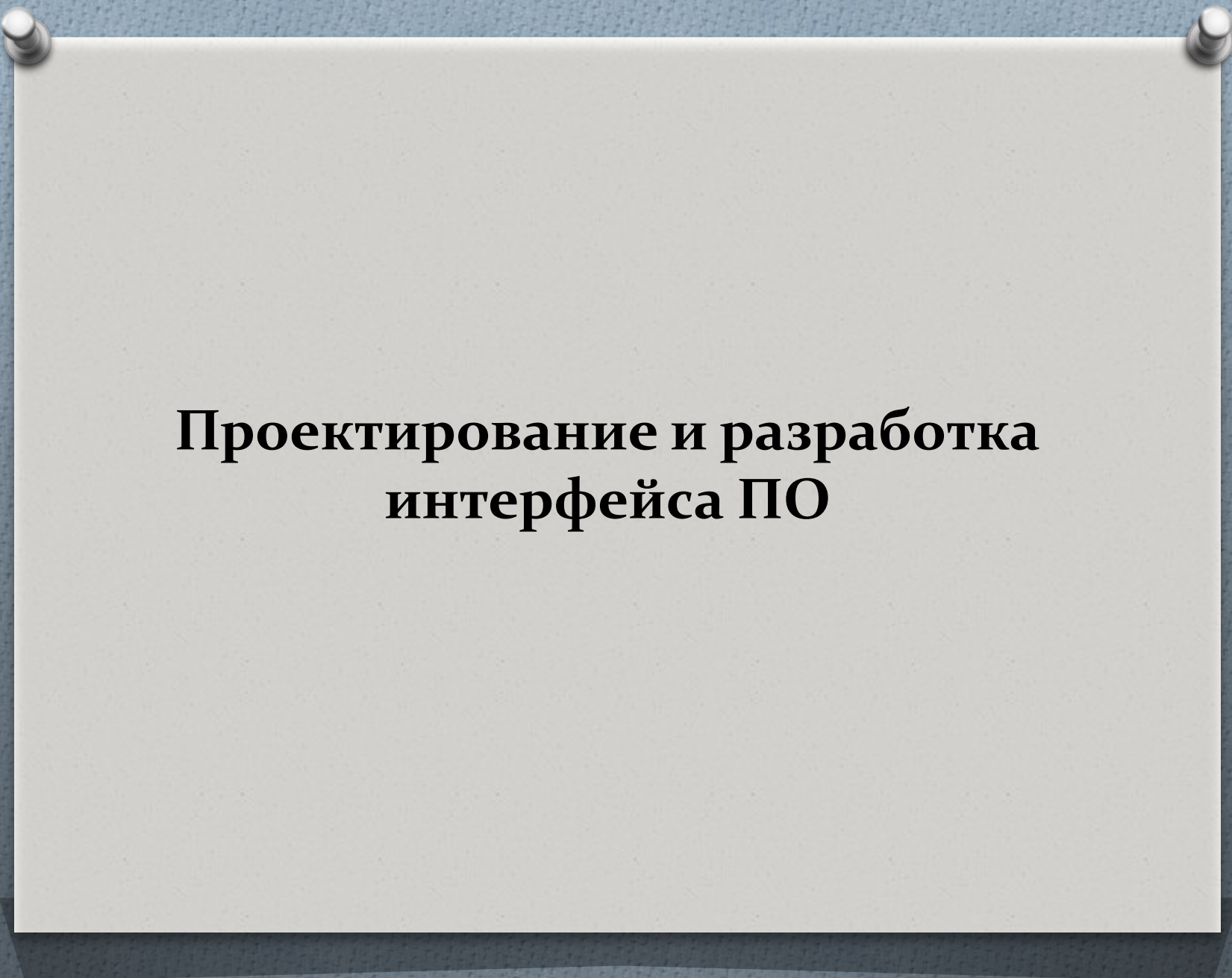
- Предварительный внешний проект
- Детальный внешний проект

Детальный внешний проект каждой функции пользователя должен включать следующую информацию:

- 0* 1. Описание входных данных.
- 0* 2. Описание выходных данных.
- 0* 3. Преобразование системы.
- 0* 4. Характеристики надежности.
- 0* 5. Эффективность.
- 0* 6. Замечания по программированию.

Чтобы изменения в проекте не приводили к дополнительным ошибкам, соблюдать следующие правила

- 0** 1. Во время всего периода работы над проектом поддерживать документацию на уровне последних решений.
- 0** 2. Фиксировать результаты каждого этапа процесса проектирования. Требования и цели фиксируются после их утверждения, внешние спецификации — после успешного завершения проверки их правильности. Изменения должны также проходить формальную процедуру утверждения.
- 0** 3. Проверять каждое внесенное изменение до такой степени, до которой проверялось исходное решение.
- 0** 4. Контролировать, чтобы нужные изменения были сделаны на всех уровнях разработки ПИ.



Проектирование и разработка интерфейса ПО

Основы построения интерфейсов

Лучший пользовательский интерфейс — это такой интерфейс, которому пользователь не должен уделять много внимания, почти не замечать его.

Программа должна помогать выполнить задачу, а не становиться этой задачей.

Программа должна работать так, чтобы пользователь не считал компьютер дураком.

При работе с программой пользователь не должен ощущать себя дураком.

Основные требования к построению интерфейса



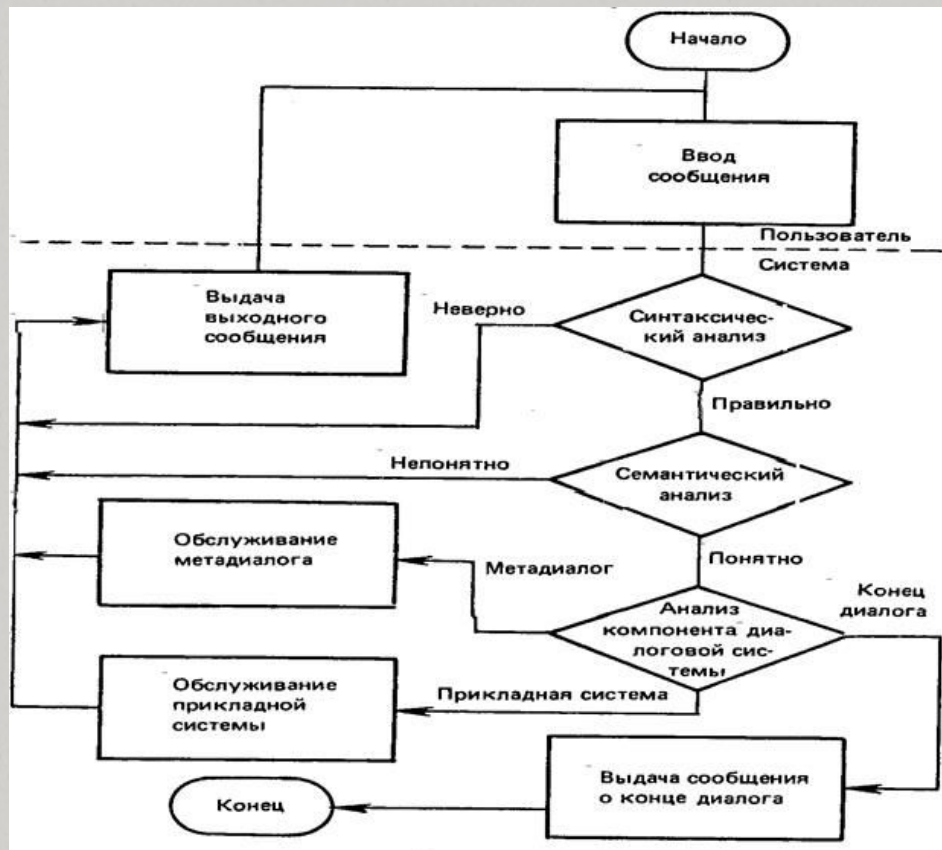
Графический
интерфейс

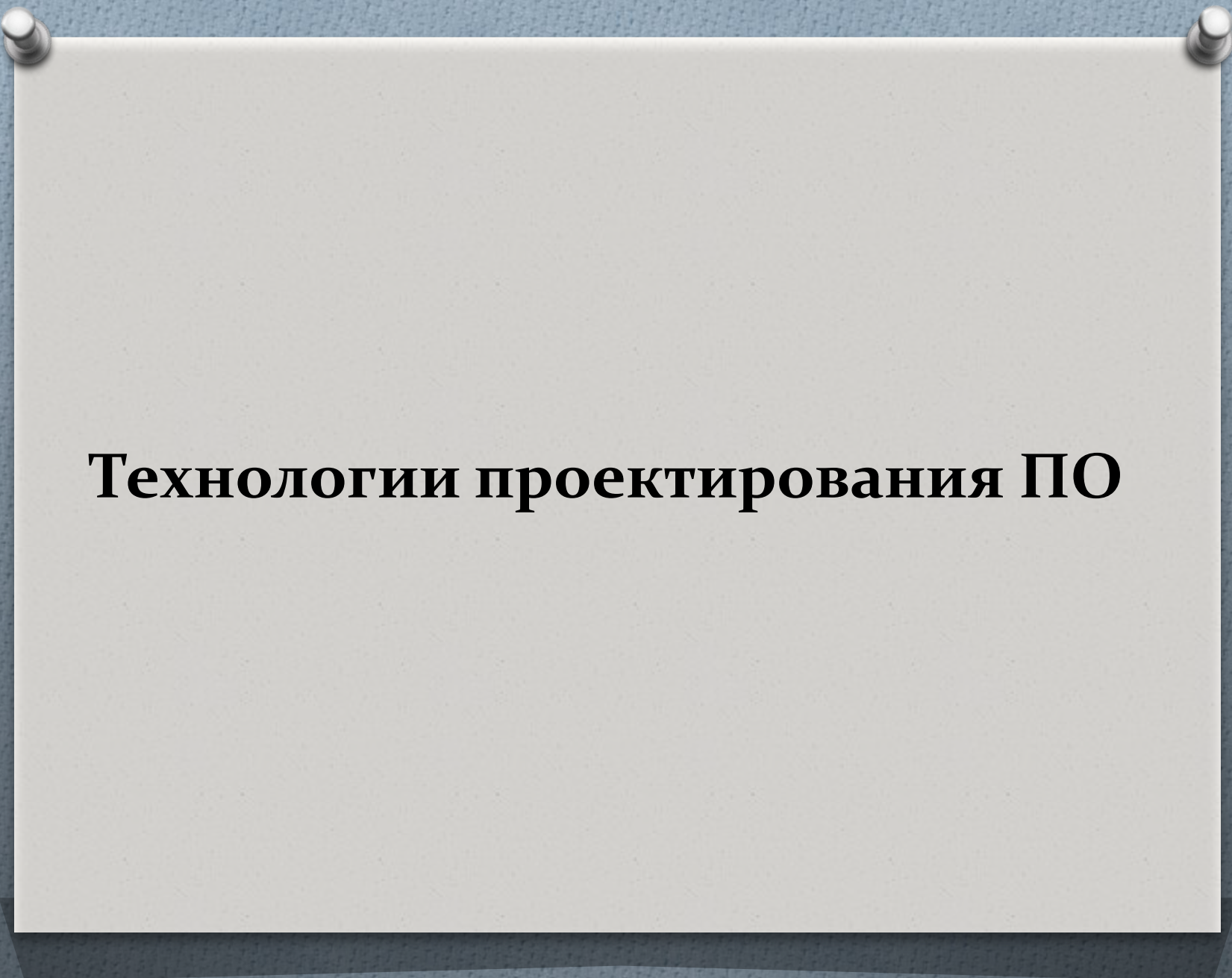


Диалоговый
режим



Описание сценария диалога





Технологии проектирования ПО

Технология разработки ПО – это совокупность процессов и методов создания программного продукта. Промышленные технологии создания программных продуктов имеют несколько обязательных характеристик, среди них:

- 0 обеспечение поддержки жизненного цикла информационной системы,
- 0 гарантия достижения целей разработки, то есть достижение требований к системе при соблюдении сроков и бюджета,
- 0 соответствие принципам управляемости,
- 0 не зависят от средств реализации ИС.

Схема технологической операции



- вывод об отсутствии адекватных технологий
- выбор конкретной технологии разработки ПО и ее приобретение;

Оценка по технико- экономическим характеристикам

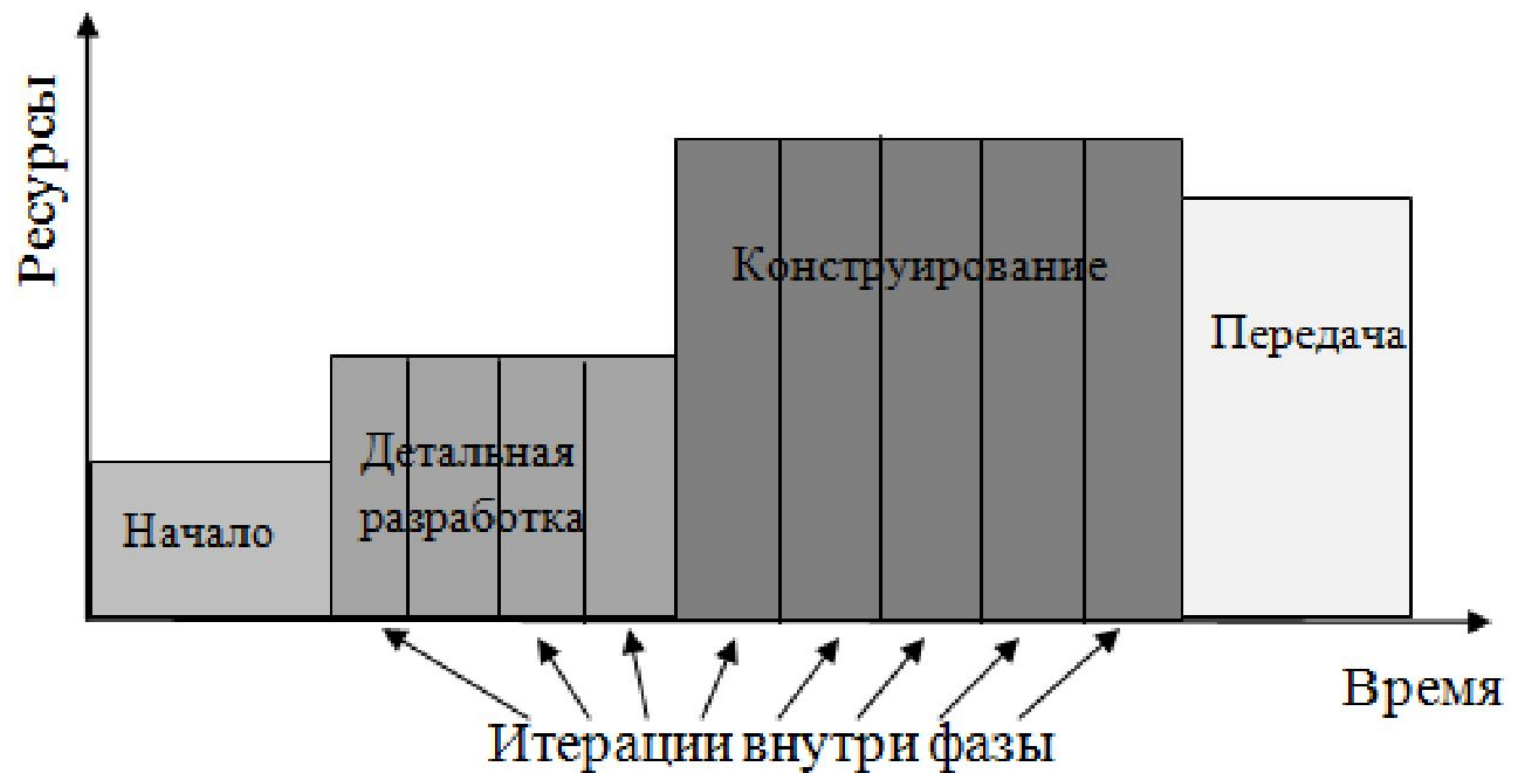
- 0 функциональные характеристики процессов жизненного цикла,
- 0 функциональные характеристики применения (среда функционирования, совместимость с другими ТС ПО, соответствие технологическим стандартам),
- 0 характеристики качества (надежность, удобство использования, эффективность, как осуществляется сопровождение, переносимость),
- 0 общие характеристики (затраты на технологию, лицензионная политика, оценочный эффект от внедрения ТС ПО, инфраструктура, требуемая для внедрения ТС ПО, доступность и качество обучения, сертификация поставщика, поддержка поставщика).

Примеры промышленных
технологий создания
программных продуктов

Rational Unified Process RUP (IBM)

- 0 Основой данной технологии является поэтапное моделирование продукта средствами UML, в ней реализуется итерационный и инкрементный подход к созданию ПО.
- 0 Разработка системы выполняется в виде нескольких краткосрочных мини-проектов фиксированной длительности (от 2 до 6 недель), называемых итерациями.
- 0 Каждая итерация включает свои собственные этапы анализа требований, проектирования, реализации, тестирования, интеграции и завершается созданием работающей системы.

Основные фазы RUP



Начало (Inception)

- 0 ☐ формируются видение и границы проекта,
- 0 ☐ создается экономическое обоснование (business case),
- 0 ☐ определяются основные требования, ограничения и ключевая функциональность продукта,
- 0 ☐ создается базовая версия,
- 0 ☐ оцениваются риски.
- 0 Результаты начальной фазы являются:
- 0 ☐ общее описание системы: основные требования к проекту, его характеристики и ограничения,
- 0 ☐ начальная модель вариантов использования (степень готовности - 10-20%),
- 0 ☐ начальный проектный глоссарий (словарь терминов),
- 0 ☐ начальный бизнес-план,
- 0 ☐ план проекта, отражающий стадии и итерации,
- 0 ☐ один или несколько прототипов.

Детальная разработка (Elaboration)

- 0 Документирование требований (включая детальное описание для большинства прецедентов использования).
- 0 Получение спроектированной, реализованной и протестированной исполняемой архитектуры.
- 0 Обновление экономического обоснования и получение более точных оценок сроков и стоимости.
- 0 Снижение основных рисков.

Результатами фазы разработки являются:

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

Построение (Construction)

- 0 итерации являются инкрементными в соответствии с той функцией, которую они выполняют. Каждая итерация добавляет очередные конструкции к вариантам использования, реализованным во время предыдущих итераций;
- 0 итерации являются повторяющимися по отношению к разрабатываемому коду. На каждой итерации некоторая часть существующего кода переписывается с целью сделать его более гибким.

Передача (Transition)

- оценивается качество продукта,
- создается финальная версия продукта.

Финальная версия передается от разработчика к заказчику (β-версия, обучение пользователей). \Фаза Передача завершается полным переводом готового ПО в эксплуатацию на стороне заказчика.

Технология RUP определяет 9 процессов, из них 6 основных и 3 вспомогательных.

Интенсивность процессов RUP на разных фазах



В результате разработки проекта с помощью *Rational Rose* формируются следующие документы:

- 0 диаграммы UML, представляющие собой модель разрабатываемой программной системы;
- 0 спецификации классов, объектов, атрибутов и операций;
- 0 заготовки текстов программ.

Custom Development Method CDM (Oracle)

- o Данная технология основана на использовании инструментального комплекса Oracle Developer Suite. Технология в основном ориентирована на разработку ПО, в котором приоритетным является разработка и использование базы данных, в том числе конверсия базы данных при переходе на новое ПО.
- o При этом технология имеет две разновидности: CDM-classic и CDM-fast track, состоит из 6-ти основных фаз и 11 процессов.

CDM classic

Определение (Definition) на данной фазе определяются цели создания системы, приоритеты и ограничения, разрабатывается системная архитектура и составляется план разработки.

Основная цель — получение согласия высшего руководства для перехода на следующую фазу.

CDM classic

Анализ (Analysis) здесь строятся модель информационных потребностей (диаграмма "сущность-связь"), диаграмма функциональной иерархии (на основе функциональной декомпозиции системы), матрица перекрестных ссылок и диаграмма потоков данных.

Основная цель — получение детальных требований к системе

CDM classic

Дизайн (Design) на данном этапе разрабатывается подробная архитектура системы, проектируются схема реляционной БД и программные модули, устанавливаются перекрестные ссылки между компонентами системы для анализа их взаимного влияния и контроля за изменениями.

Основная цель — перевод требований в спецификации.

CDM classic

Построение (Build). При построении создается БД, строятся прикладные системы, производится их тестирование, проверка качества и соответствия требованиям пользователей. Создается системная документация, материалы для обучения и руководства пользователей.

Основная цель — получение продукта.

CDM classic

Передача (Transition) здесь анализируются производительность и целостность системы.

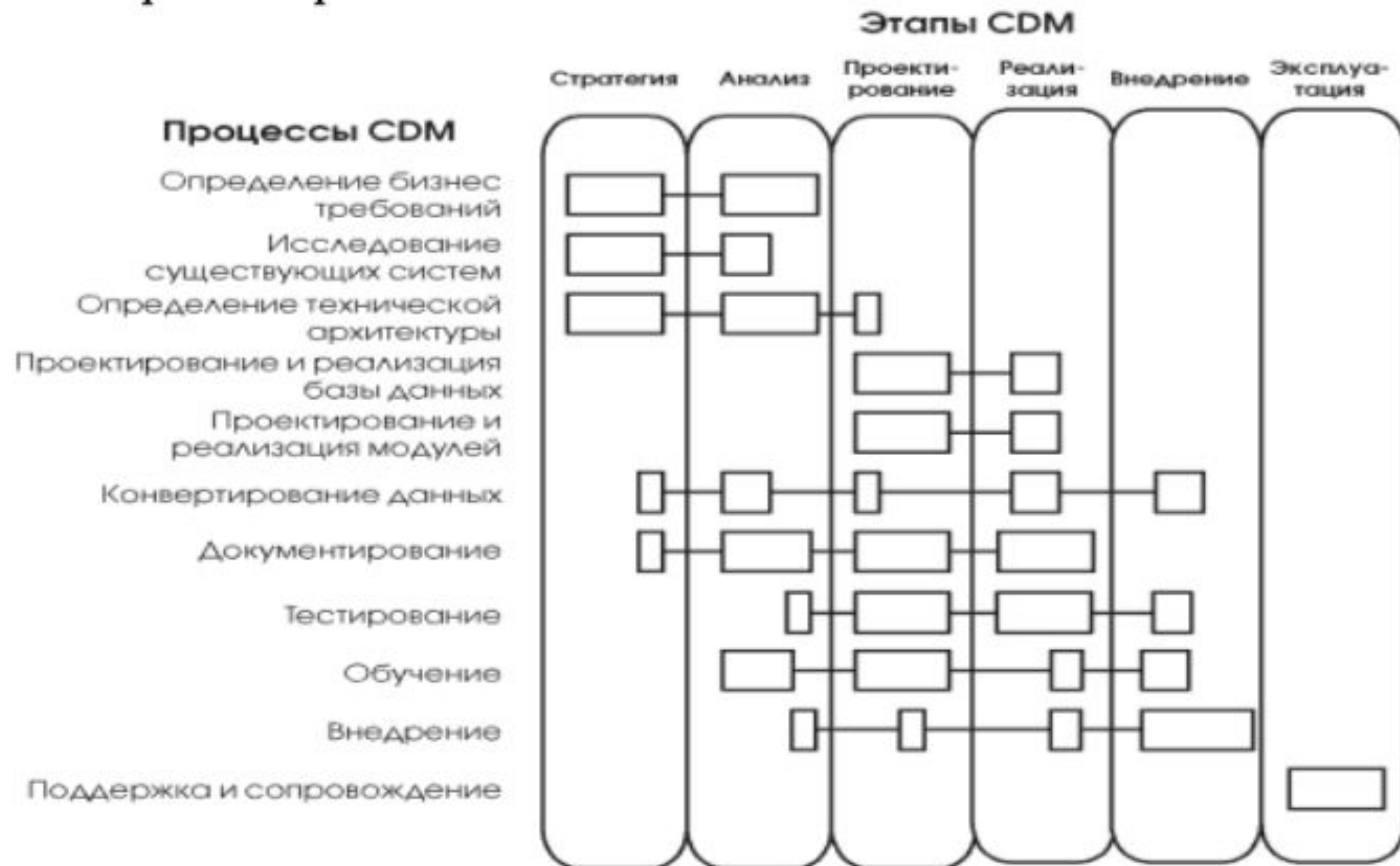
*Основная цель — установка системы у клиента и подготовка к стадии *production*.*

CDM classic

Работа (Production) на данной фазе выполняется поддержка и, при необходимости, модификация системы.

Основная цель — поддержание работоспособности системы.

Распределение процессов по фазам

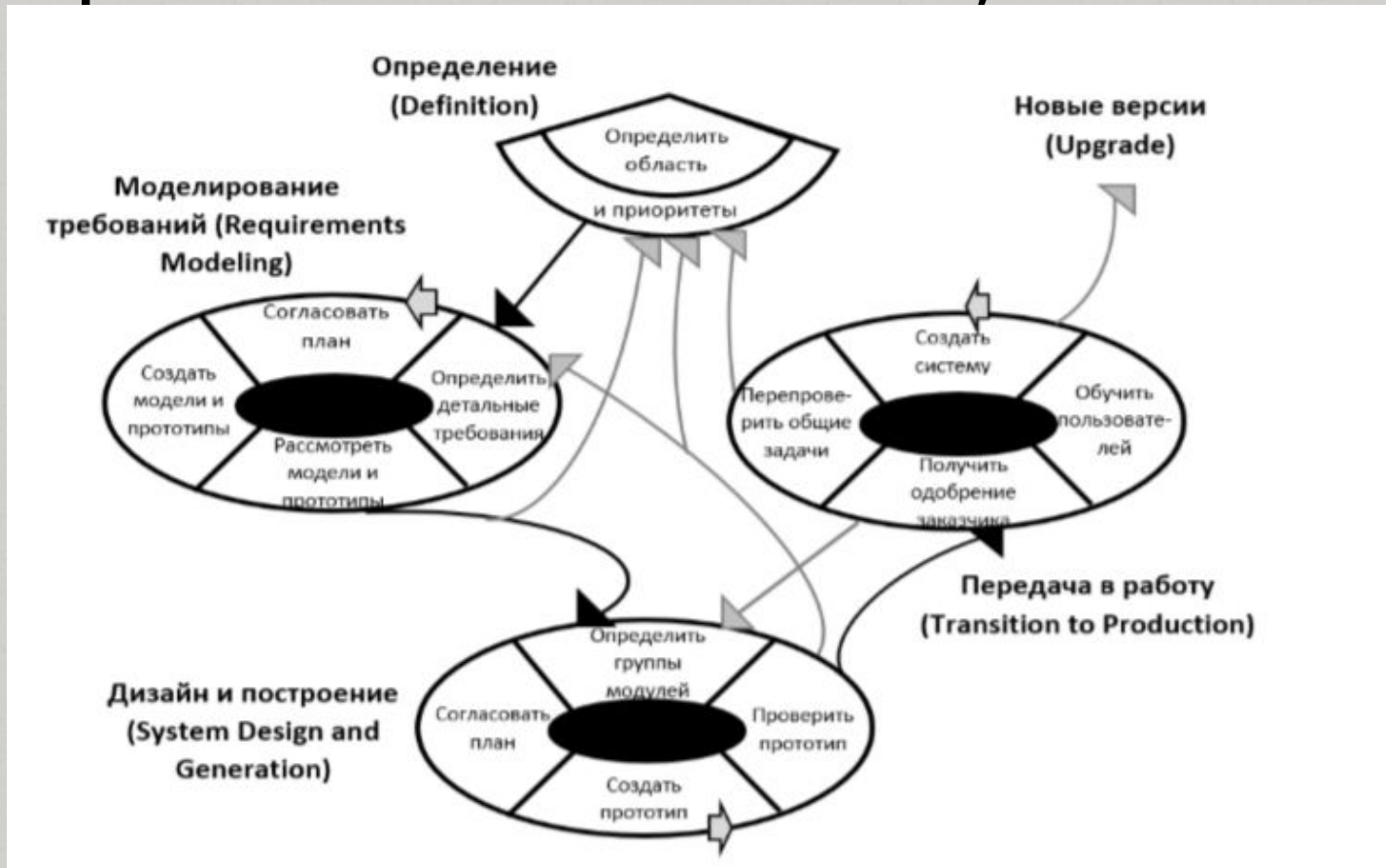


CDM fast track

Ориентация на быстрое получение продукта, следовательно, разработка имеет короткий цикл и должна содержать небольшой объем работ.

Данная технология, благодаря своим характеристикам, входит в группу *Agile* (*подвижный, сообразительный*) методологий.

Графическое представление фаз технологии *CDM fast trac*



Microsoft Solution Framework

MSF (MicroSoft)

- 0 Технология MSF состоит из 4х этапов, каждый из которых завершается "вехой" – ключевой точкой, в которой производится оценка достигнутого, каждый из этапов сопровождается группой из 6и процессов, в зависимости от этапа в процессе смещается точка фокуса.
- 0 Веха каждого из этапов – это определенные задачи с необходимым уровнем качества их выполнения. После достижения заданного уровня, можно переходить на выполнение следующего этапа.

Выработка концепции (Envisioning)

Промежуточные задачи этапа:

- 0 оценка существующей ситуации;
- 0 определение состава команды;
- 0 определение структуры проекта;
- 0 определение бизнес-целей;
- 0 определение требований и профилей пользователей;
- 0 разработка концепции решения;
- 0 создание документа общей картины и области действия проекта;
- 0 оценка риска.

Промежуточные вехи:

- 0 Организован костяк команды;
- 0 Создана общая картина решения.

Финальная веха:

- 0 Утверждение документа общей картины.

Планирование (Panning)

Промежуточные задачи этапа:

- 0 анализ и документирование требований;
- 0 разработка проект и основные архитектурные решения;
- 0 функциональные спецификации системы;
- 0 планы и календарные графики (тестирования и пилотной эксплуатации);
- 0 выбор среды разработки.

Промежуточные вехи или стадии проектирования:

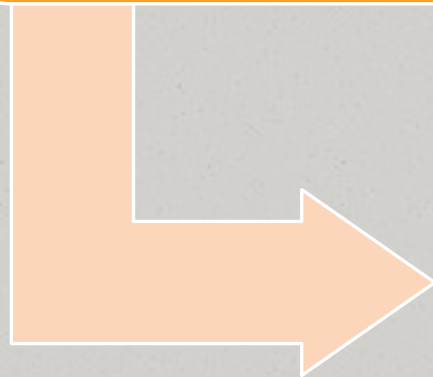
- 0 концептуальное,
- 0 логическое,
- 0 физическое.

Финальная веха:

- 0 утверждение проектных планов.

Планирование (Panning)

Генеральный
план и
календарный
график проект



План
управления
рискам

Разработка (Developing)

Промежуточные задачи этапа:

- 0 создание компонент решения (документация, код);
- 0 разработка инфраструктуры.

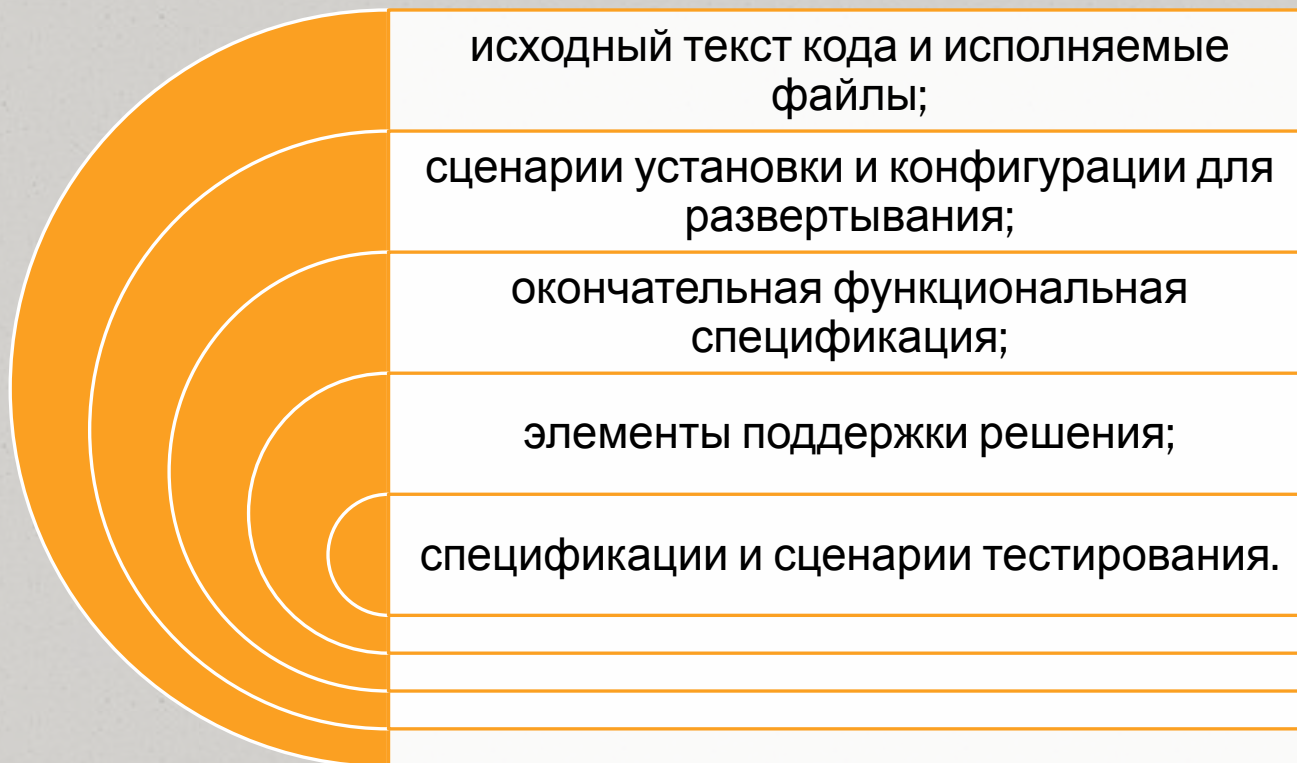
Промежуточные вехи – разработка завершена:

- 0 решение готово к тестированию и стабилизации;
- 0 выявление всех оставшихся проблем.

Финальная веха:

- 0 окончательное утверждение области действия проекта.

Результаты этапа предполагают следующие элементы:



Стабилизация (Stabilizing)

Промежуточные задачи этапа:

- 0 подготовка к выпуску окончательной версии продукта;
- 0 доведение до заданного уровня качества;
- 0 определение состава команды;
- 0 обнаружение и устранение дефектов;
- 0 пилотная эксплуатация в тестовой среде.

Финальная веха:

- 0 подтверждение готовности проекта к выпуску.

Развертывание (Deploying)

Промежуточные задачи этапа:

- 0* установка решения и необходимых компонентов окружения;
- 0* стабилизация в промышленных условиях;
- 0* передача проекта в руки группы сопровождения;
- 0* анализ проекта в целом на предмет уровня удовлетворенности заказчика.

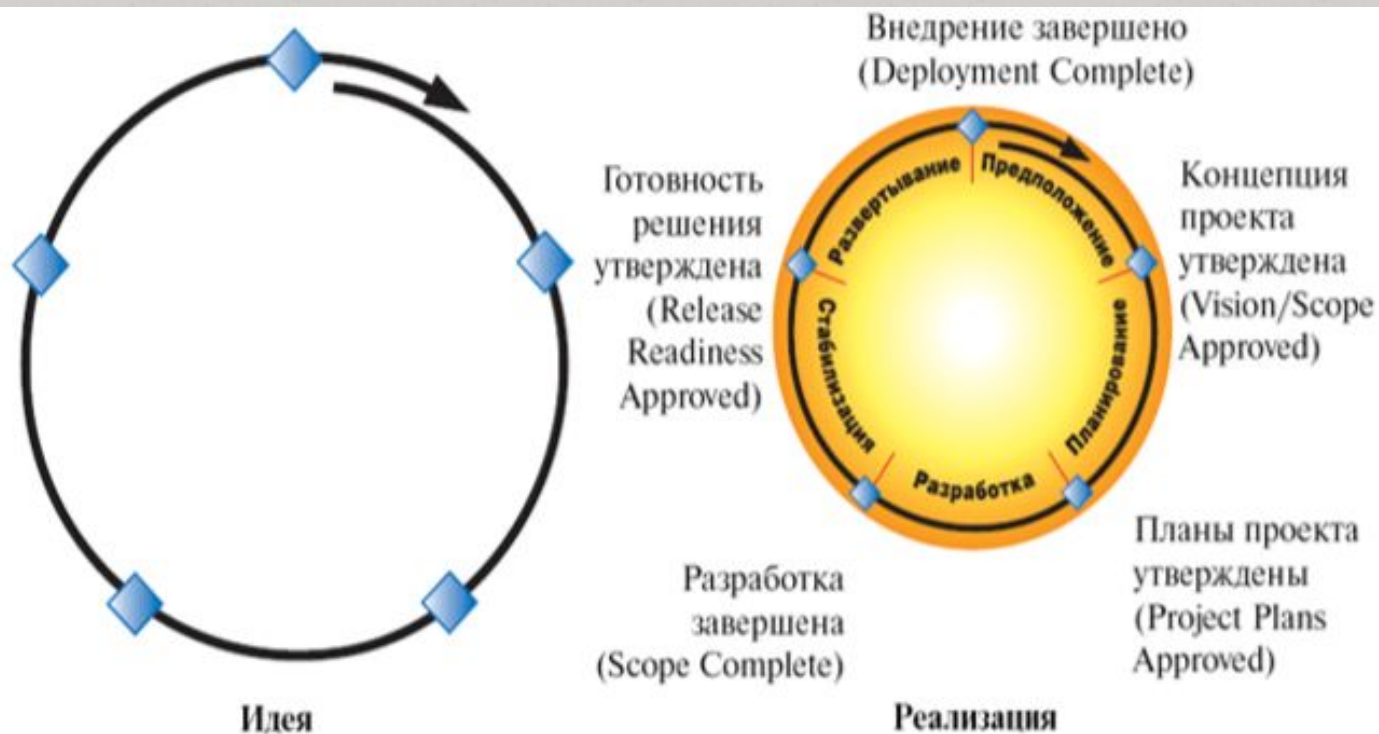
Финальная веха:

- 0* Подтверждение завершения внедрения .

Технология MSF определяет следующие процессы, сопровождающие создание программного продукта



Графическое представление фаз технологии MSF, этапы и финальные вехи



Extreme Programming XP

Основными или общими характеристиками данной группы технологий является:

- получение быстрых результатов в малом проекте с ограниченными ресурсами;
- малая рабочая группа (до 50 человек);
- короткий цикл разработки (до 6 месяцев).

Extreme Programming XP

Планирование

- Осуществляется на основе бизнес-приоритетов заказчика и технических возможностей
- Сбор/ Отбор User Story
- Определяется время окончания версии

Дизайн

- Вброс архитектуры
- Выбор технологий
- Создание metaphor системы

Кодирование

- Только отобранные User Stories
- Осуществляется в парах (2 программиста на одном компьютере)
- Максимальная простота кода

Тестирование

Тесты на основании User Story (создаются на этапе планирования)
Вместе с заказчиком

Графическое представление фаз технологии XR

