Модели жизненного цикла программного обеспечения

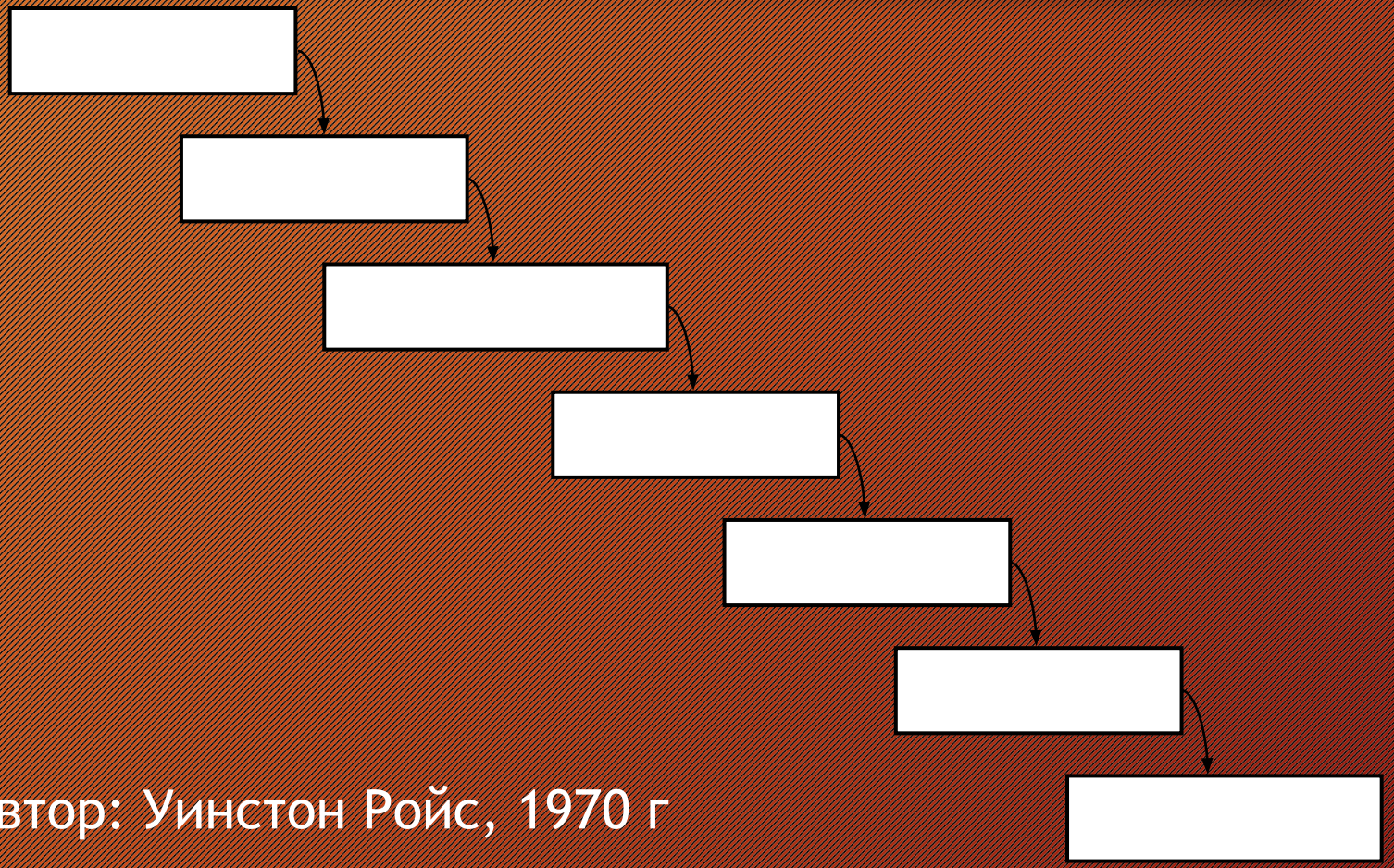
Шарифкулов Марат Хаджи-Муратович
гр.2-1П11

Модели жизненного цикла

Основные модели жизненного цикла ПО:

- Каскадная модель
- Макетирование
- Инкрементная модель
- Спиральная модель

Каскадная (водопадная) модель



Автор: Уинстон Ройс, 1970 г

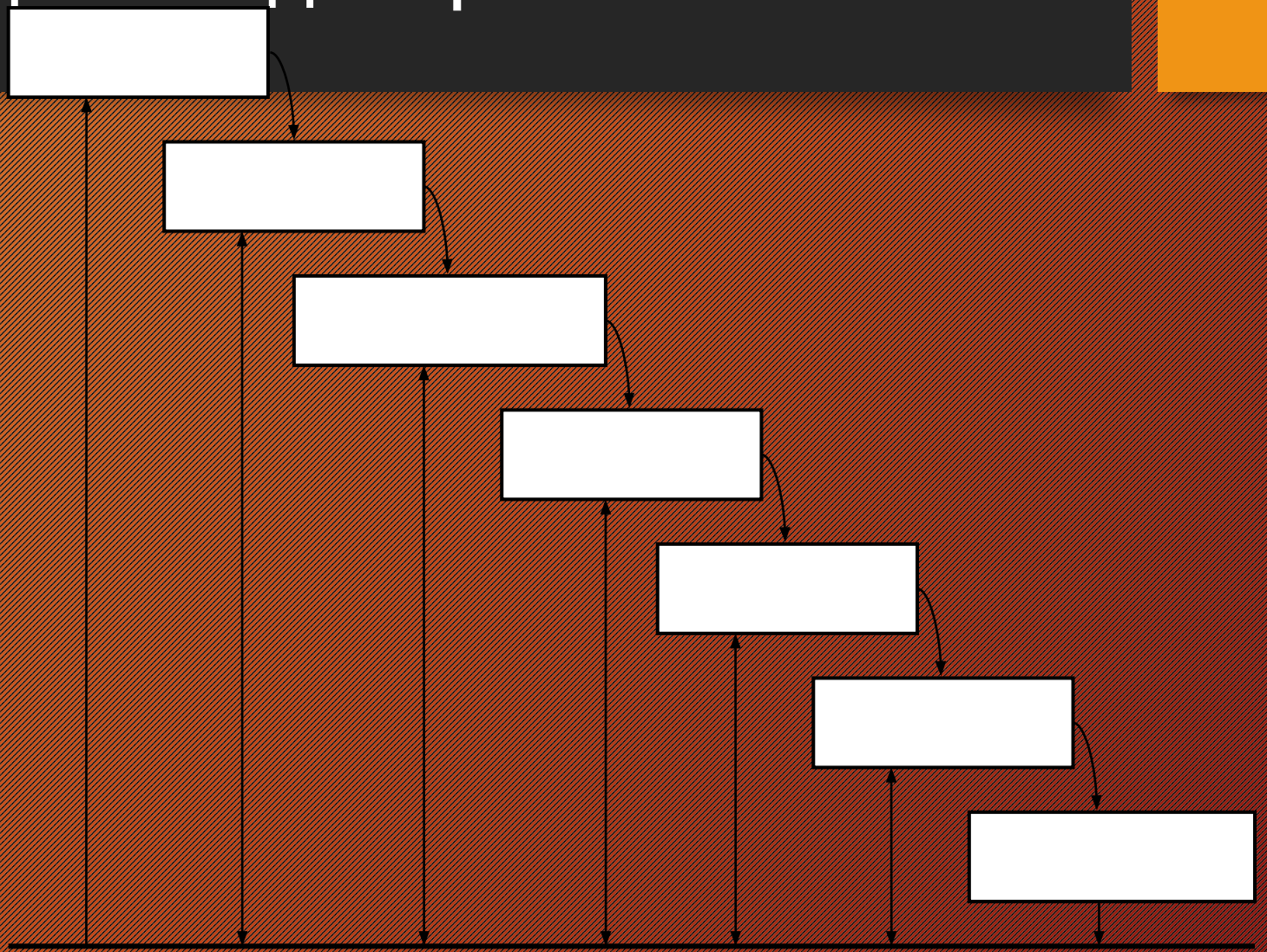
Преимущества каскадной модели

- широкая известность и простота модели;
- упорядоченность преодоления сложностей и хорошо срабатывает для тех проектов, которые достаточно понятны, но все же трудно разрешимы;
- отличается стабильностью требований;
- удобна, когда требования к качеству доминируют над требованиями к затратам и графику выполнения проекта;
- способствует осуществлению строгого контроля менеджмента проекта;
- позволяет участникам проекта, завершившим действия на выполняемой ими фазе, принять участие в реализации других проектов;
- определяет процедуры по контролю за качеством;
- стадии модели довольно хорошо определены и понятны, легко отслеживаются с помощью временной шкалы или графика Гантта.

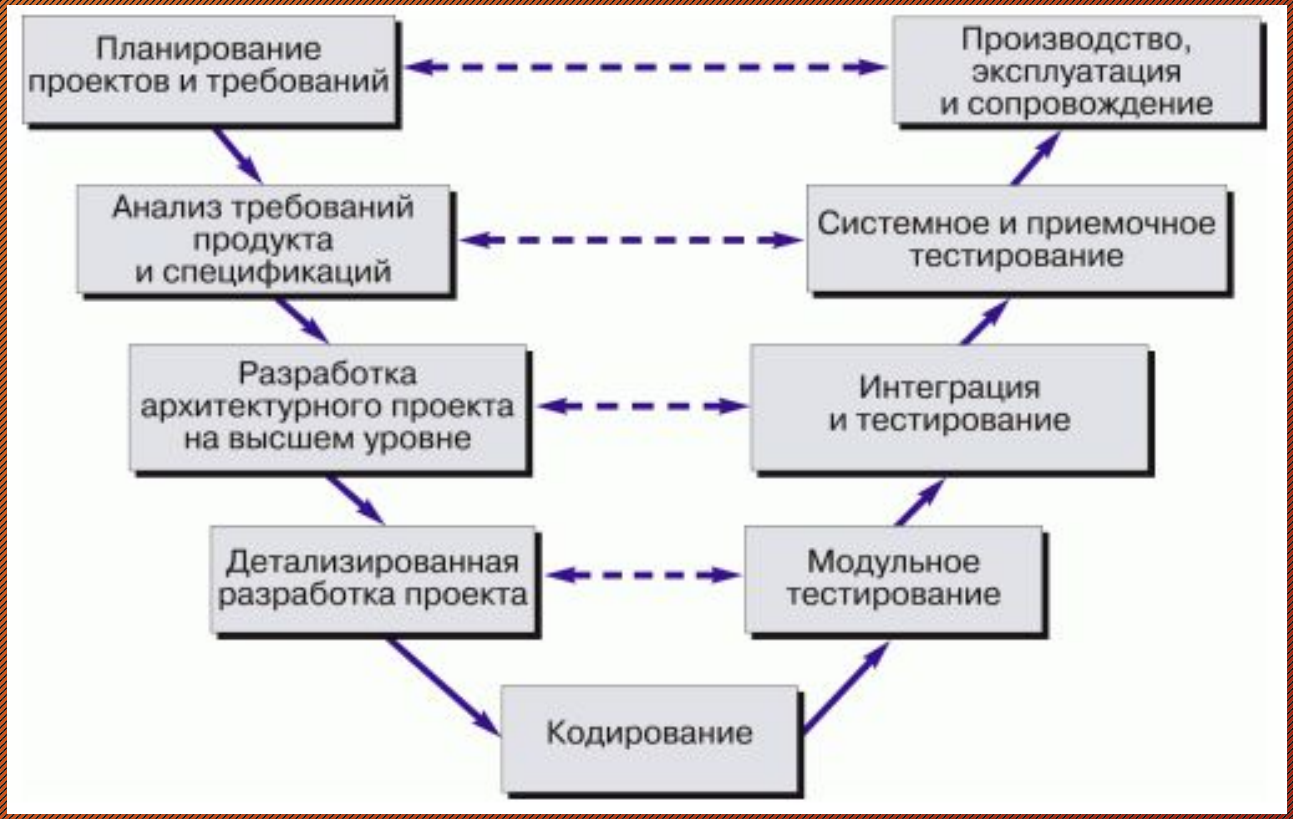
Недостатки каскадной модели

- в основе модели лежит последовательная линейная структура;
- невозможность предотвращения возникновение итераций между фазами;
- она может создать ошибочное впечатление о работе над проектом;
- интеграция всех полученных результатов происходит внезапно в завершающей стадии работы модели и у клиента практически нет возможности ознакомиться с системой заранее;
- каждая фаза является предпосылкой для выполнения последующих действий и ее результат считается замороженным;
- все требования должны быть известны в начале жизненного цикла;
- необходимость в жестком управлении и контроле;
- модель основана на документации;
- весь программный продукт разрабатывается за один раз;
- отсутствует возможность учесть переделку и итерации за рамками проекта.

Модель водоворота



V-образная модель



Преимущества V-образной модели

- в модели особое значение придается планированию, направленному на верификацию и аттестацию разрабатываемого продукта на ранних стадиях его разработки;
- в модели предусмотрены аттестация и верификация всех внешних и внутренних полученных данных, а не только самого программного продукта;
- в V-образной модели определение требований выполняется перед разработкой проекта системы, а проектирование ПО – перед разработкой компонентов;
- модель определяет продукты, которые должны быть получены в результате процесса разработки;
- благодаря модели менеджеры проекта может отслеживать ход процесса разработки, так как в данном случае вполне возможно воспользоваться временной шкалой, а завершение каждой фазы является контрольной точкой;
- модель проста в использовании.

Недостатки V-образной модели

- с ее помощью непросто справиться с параллельными событиями;
- в ней не учтены итерации между фазами;
- в модели не предусмотрено внесение требования динамических изменений на разных этапах жизненного цикла;
- тестирование требований в жизненном цикле происходит слишком поздно, вследствие чего невозможно внести изменения, не повлияв при этом на график выполнения проекта;
- в модель не входят действия, направленные на анализ рисков.

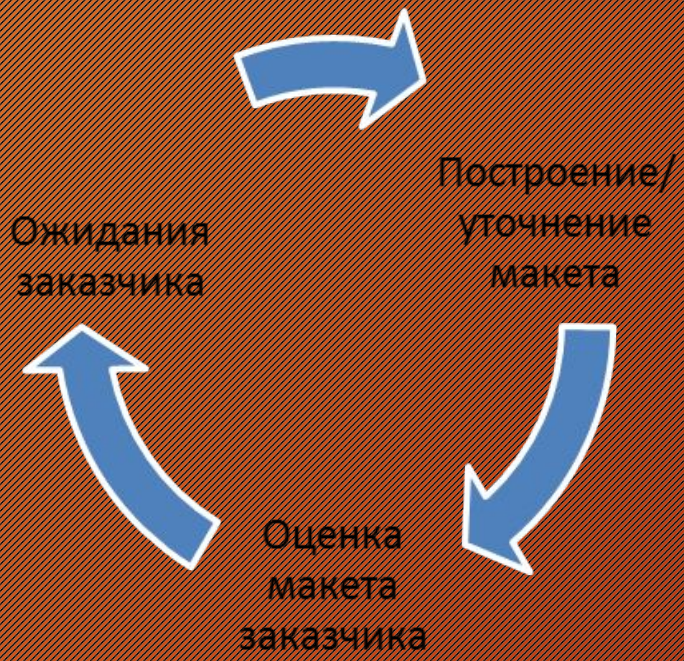
Макетирование

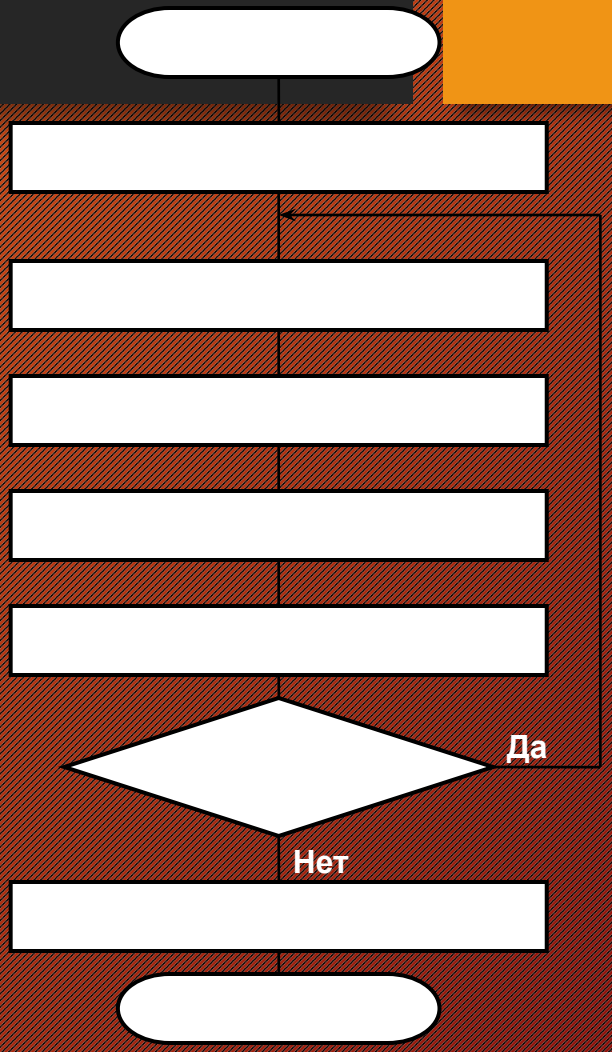
Макетирование (прототипирование) – это процесс создания модели разрабатываемого программного продукта.

Модель может принимать один из трех видов:

- бумажный макет или «электронный» макет, который представляет человеко-машинный интерфейс;
- работающий макет (выполняет только часть требуемых функций);
- существующая программа (характеристики которой должны быть улучшены).

Процесс макетирования





Преимущества макетирования

- конечный пользователь может "увидеть" системные требования в процессе их сбора командой разработчиков;
- снижается возможность возникновения путаницы, искажения информации или недоразумений при определении системных требований;
- возможность внесения новых или неожиданных требований пользователя;
- минимизация возникновения разногласий при общении заказчиков с разработчиками;
- модель позволяет выполнять гибкое проектирование и разработку;
- образуются постоянные, видимые признаки прогресса в выполнении проекта;

Преимущества макетирования

- принимая участие в процессе разработки на протяжении всего ЖЦ, пользователи в большей степени будут довольны полученными результатами;
- ожидаемое качество продукта определяется при активном участии пользователя в процессе на ранних фазах разработки;
- благодаря меньшему объему доработок уменьшаются затраты на разработку;
- обеспечивается управление рисками;
- документация сконцентрирована на конечном продукте, а не на его разработке.

Недостатки макетирования

- требуется активное участие заказчика;
- на заказчиков может оказать негативное влияние тот факт, что они не располагают информацией о точном количестве итераций, которые будут необходимы;
- заказчик может предпочесть получить прототип, вместо того, чтобы ждать появления полной, хорошо продуманной версии;
- с учетом создания рабочего прототипа, качеству всего ПО или долгосрочной эксплуатационной надежности может быть уделено недостаточно внимания.
- прототипирование вызывает зависимость и может продолжаться слишком долго;
- при использовании модели решение трудных проблем может отодвигаться на будущее;
- при выборе инструментальных средств прототипирования (операционные системы, языки и малопродуктивные алгоритмы) разработчики могут остановить свой выбор на менее подходящем решении, только чтобы продемонстрировать свои способности.

Инкрементная модель жизненного цикла

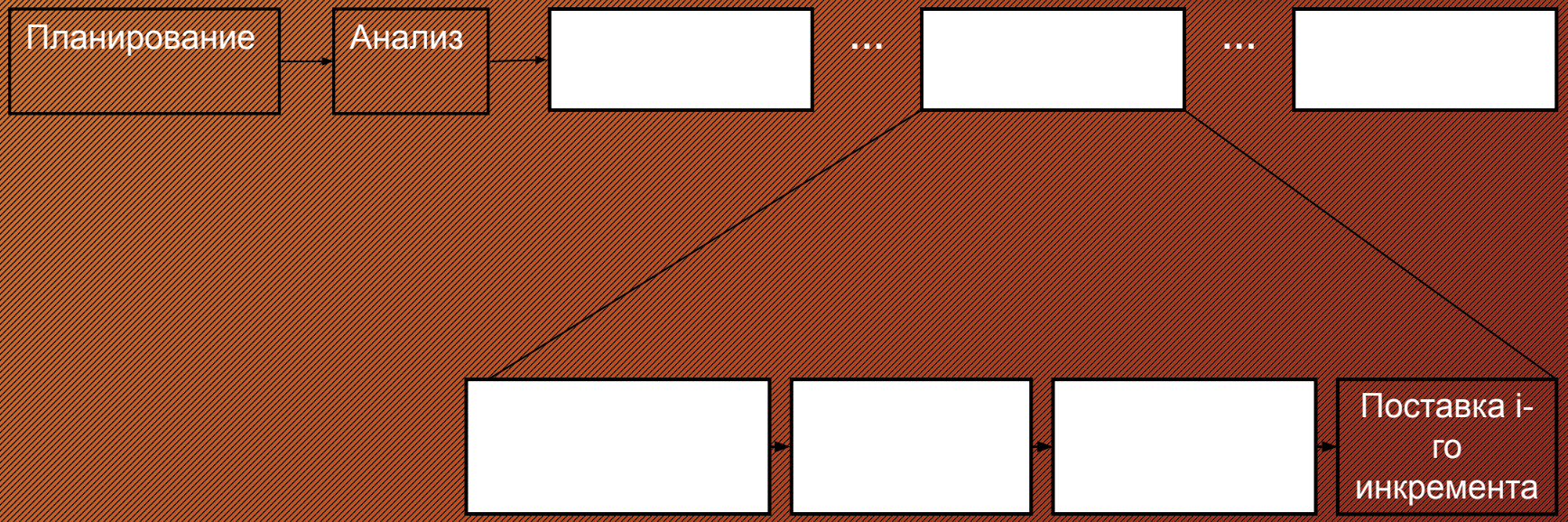
Инкрементная разработка представляет собой процесс частичной реализации всей системы и медленного наращивания функциональных возможностей.

Инкрементная модель действует по принципу каскадной модели с перекрытиями.

Два подхода к набору требований:

- полный заранее сформированный набор требований, которые выполняются в виде последовательных, небольших по размеру проектов,
- выполнение проекта может начаться с формулирования общих целей, которые затем уточняются и реализуются группами разработчиков.

Инкрементная модель ЖЦ



Преимущества инкрементной модели

- в результате выполнения каждого инкремента получается функциональный продукт;
- заказчик располагает возможностью высказаться по поводу каждой разработанной версии системы;
- правило по принципу "разделяй и властвуй" позволяет разбить возникшую проблему на управляемые части;
- заказчики могут распознавать самые важные и полезные функциональные возможности продукта на более ранних этапах разработки;
- требования стабилизируются на момент создания определенного инкремента;
- инкременты функциональных возможностей несут больше пользы и проще при тестировании
- снижается риск неудачи и изменения требований;
- риск распределяется на несколько меньших по размеру инкрементов;
- существует возможность пересмотреть риски, связанные с затратами и соблюдением установленного графика;
- заказчик может привыкать к новой технологии постепенно.

Недостатки инкрементной модели

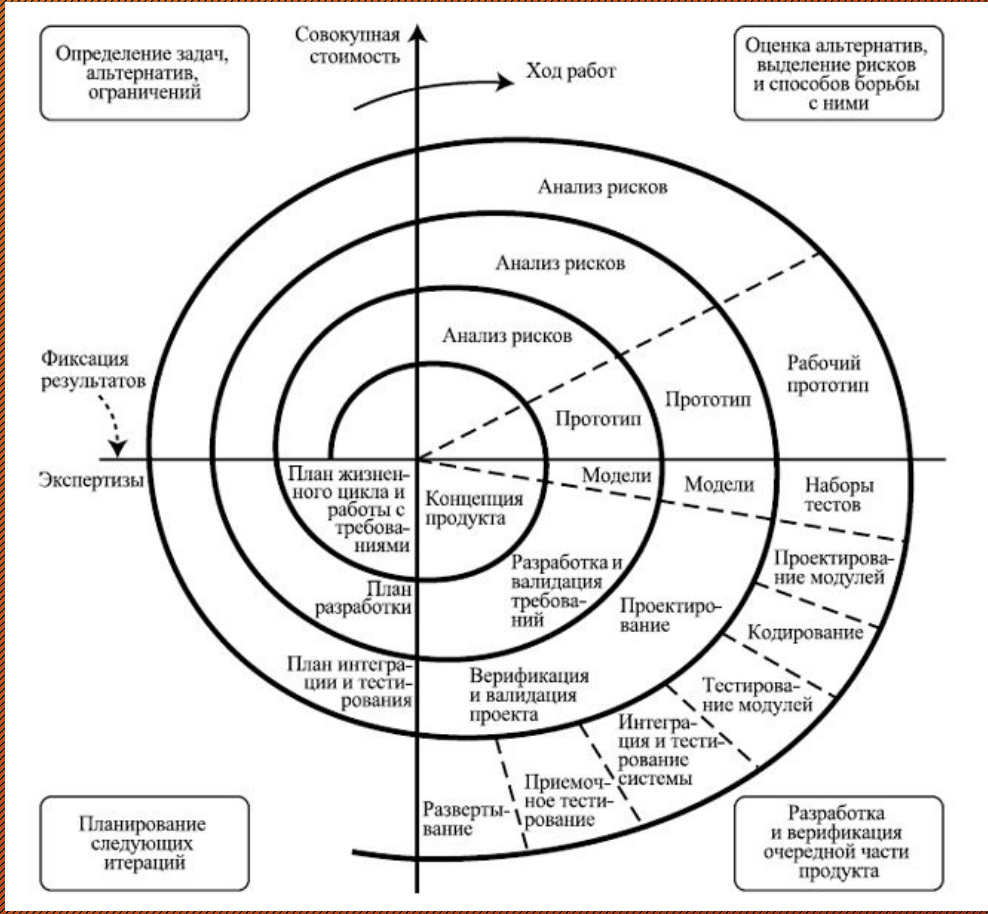
- определение полной функциональной системы должно осуществляться в начале ЖЦ, чтобы обеспечить определение инкрементов;
- для модели необходимы хорошее планирование и проектирование;
- использование на этапе анализа общих целей, вместо полностью сформулированных требований, может оказаться неудобным для руководства;
- поскольку создание некоторых модулей будет завершено значительно раньше других, возникает необходимость в четко определенных интерфейсах;
- заказчик должен осознавать, что общие затраты на выполнение проекта не будут снижены;
- в модели не предусмотрены итерации в рамках каждого инкремента;
- формальный критический анализ и проверку намного труднее выполнить для инкрементов, чем для системы в целом;
- может возникнуть тенденция к оттягиванию решений трудных проблем на будущее с целью продемонстрировать руководству успех, достигнутый на ранних этапах разработки.

Спиральная модель

Спиральная модель (автор: Барри Боэм, 1988) является реализацией эволюционной стратегии разработки программного обеспечения.

Спиральная модель отображает базовую концепцию, которая заключается в том, что каждый цикл представляет собой набор операций, которому соответствует такое же количество стадий, как и в модели каскадного процесса.

Спиральная модель



Преимущества спиральной модели

- позволяет пользователям "увидеть" систему на ранних этапах;
- она обеспечивает разбиение большого потенциального объема работы по разработке продукта на небольшие части;
- в модели предусмотрена возможность гибкого проектирования;
- реализованы преимущества инкрементной модели (выпуск инкрементов, сокращение графика посредством перекрывания инкрементов);
- быстрая обратная связь по направлению от пользователей к разработчикам;
- обеспечивается определение непреодолимых рисков без особых дополнительных затрат;
- эта модель разрешает пользователям активно принимать участие при планировании, анализе рисков, разработке, а также при выполнении оценочных действий;
- при использовании спиральной модели не нужно распределять заранее все необходимые для выполнения проекта финансовые ресурсы.

Недостатки спиральной модели

- модель имеет усложненную структуру, поэтому может быть затруднено ее применение разработчиками, менеджерами и заказчиками;
- спираль может продолжаться до бесконечности, поскольку каждая ответная реакция заказчика на созданную версию может порождать новый цикл;
- могут возникнуть затруднения при определении целей и стадий, указывающих на готовность продолжать процесс разработки на следующей итерации;
- большое количество промежуточных стадий может привести к необходимости в обработке внутренней дополнительной и внешней документации;
- если проект имеет низкую степень риска или небольшие размеры, модель может оказаться дорогостоящей;
- отсутствие хорошего средства или метода прототипирования может сделать использование модели неудобным.

Компонентная модель